

# Glyphs Handbook



May 2013

You are reading the Glyphs Handbook from May 2013.  
Please download the latest version at:  
**[glyphsapp.com/getting-started/tutorials/](http://glyphsapp.com/getting-started/tutorials/)**

© 2011–2013 glyphsapp.com

Written by Rainer Erich Scheichelbauer and Georg Seifert.

Thanks to Jeff Kellem, Rob Keller, Toshi Omagari and  
Claus Eggert Sørensen for their invaluable input.

# Contents

- 1 Glyphs 6**
  - 1.1 A Tool for Creating Fonts 6
  - 1.2 System Requirements 6
  - 1.3 Community 6
  - 1.4 Updates 6
  - 1.5 Glyphs Mini 7
  - 1.6 Keyboard Shortcuts 7
  
- 2 Edit View 8**
  - 2.1 Drawing Paths 8
    - 2.1.1 Freehand Outlines 8
    - 2.1.2 Primitives 8
  - 2.2 Editing Paths 9
    - 2.2.1 Selecting, Moving, Converting 9
    - 2.2.2 Scaling and Rotating 10
    - 2.2.3 Aligning 10
    - 2.2.4 Deleting Nodes 11
    - 2.2.5 Opening and Closing Paths 12
    - 2.2.6 Cutting Paths 12
    - 2.2.7 Resegmending Outlines 12
    - 2.2.8 Controlling Path Direction 13
    - 2.2.9 Extremes and Inflection Points 14
  - 2.3 Guidelines 14
    - 2.3.1 Magnetic Guidelines 14
    - 2.3.2 Local and Global Guidelines 14
  - 2.4 Glyph Display 15
    - 2.4.1 Zooming 15
    - 2.4.2 Panning 16
    - 2.4.3 Nodes in Alignment Zones 16
    - 2.4.4 Components, Anchors, Clouds 17
    - 2.4.5 Special Marks for Glyph Composition 20
    - 2.4.6 Mark to Base Positioning 20
    - 2.4.7 Mark to Mark Positioning 20
    - 2.4.8 Cursive Attachment Anchors 21
  - 2.5 Background 21
  - 2.6 Entering Text 21
    - 2.6.1 Sample Texts 22
    - 2.6.2 Text Tool 22
    - 2.6.3 Writing Direction 23
  - 2.7 Measuring 23
    - 2.7.1 Info Box 23
    - 2.7.2 Measurement Tool 25
    - 2.7.3 Measurement Guidelines 25
    - 2.7.4 Measurement Line 26
  - 2.8 Annotating 27
    - 2.8.1 Annotation Cursor 27
    - 2.8.2 Annotation Text 27
    - 2.8.3 Annotation Arrow 28
    - 2.8.4 Annotation Circle 28
    - 2.8.5 Plus and Minus Annotations 28
  - 2.9 Images 28
    - 2.9.1 Adding Images 28
    - 2.9.2 Manipulating Images 29
  - 2.10 Previewing and Testing 29
    - 2.10.1 Previewing Kerning 29
    - 2.10.2 Previewing Masters 29
    - 2.10.3 Previewing Path Offset 30
    - 2.10.4 Previewing OpenType Features 30
    - 2.10.5 Previewing Interpolated Instances 30
    - 2.10.6 Previewing in OS X 31
    - 2.10.7 Previewing in Adobe applications 32
  - 2.11 Exchanging Outlines with Illustrator 32
    - 2.11.1 Preparation 32
    - 2.11.2 Set the Page Origin 32
    - 2.11.3 Copy the Paths 33

- 3 Palette 34**
  - 3.1 Palette Window 34
  - 3.2 Dimensions 34
  - 3.3 Fit Curve 34
  - 3.4 Layers 35
    - 3.4.1 Working with Layers 36
    - 3.4.2 Switching Glyph Shapes 36
  - 3.5 Transformations 37
- 4 Filters 39**
  - 4.1 Filters Menu 39
  - 4.2 Hatch Outline 39
  - 4.3 Offset Curve 39
  - 4.4 Remove Overlap 40
  - 4.5 Roughen 40
  - 4.6 Round Corners 40
  - 4.7 Transformations 41
    - 4.7.1 Transform 41
    - 4.7.2 Background 41
    - 4.7.3 Metrics 41
  - 4.8 Third-Party Filters 41
- 5 Font View 42**
  - 5.1 Viewing Glyphs 42
    - 5.1.1 Grid View 42
    - 5.1.2 List View 43
  - 5.2 Adding Glyphs 43
    - 5.2.1 Generating New Glyphs 43
    - 5.2.2 Copying Glyphs Between Files 44
  - 5.3 Glyph Properties 45
    - 5.3.1 Name 45
    - 5.3.2 Width and Sidebearings 46
    - 5.3.3 Kerning Groups 46
    - 5.3.4 Exports 46
    - 5.3.5 Color Label 46
    - 5.3.6 Unicode 46
    - 5.3.7 Note 46
    - 5.3.8 Components in List View 46
    - 5.3.9 Read-Only Properties in List View 47
  - 5.4 Batch-Processing 47
    - 5.4.1 Layer Commands and Filters 47
    - 5.4.2 Palette Manipulations 47
  - 5.5 Filtering and Sorting 48
    - 5.5.1 Search Box 48
    - 5.5.2 Categories 48
    - 5.5.3 Languages 48
    - 5.5.4 Custom Filters 49
    - 5.5.5 List Filters 49
    - 5.5.6 Custom Parameter 'glyphOrder' 50
  - 5.6 Names and Unicode 50
    - 5.6.1 Readable or Nice Names 50
    - 5.6.2 Naming Glyphs 50
    - 5.6.3 Batch Renaming 52
  - 5.7 Images 52
    - 5.7.1 Adding Images 52
    - 5.7.2 Viewing Images 52
- 6 Font Info 53**
  - 6.1 Font 53
    - 6.1.1 Family Name 53
    - 6.1.2 Units per Em 53
    - 6.1.3 Designer and Designer URL 53
    - 6.1.4 Manufacturer and Manufacturer URL 54
    - 6.1.5 Copyright 54
    - 6.1.6 Version 54
    - 6.1.7 Date 54
    - 6.1.8 Custom Parameters 54
  - 6.2 Masters 55
    - 6.2.1 Proportions: Width and Weight 55
    - 6.2.2 Metrics 55
    - 6.2.3 Stems 56
    - 6.2.4 Alignment Zones 56
    - 6.2.5 Custom Parameters 57
  - 6.3 Instances 57
    - 6.3.1 Style Name 57
    - 6.3.2 Weight and Width 57
    - 6.3.3 Style Linking 58
    - 6.3.4 Interpolation 58

- 6.3.5 Custom Parameters 58
- 6.4 Features 60
- 6.5 Other Settings 61
  - 6.5.1 Grid Spacing 61
  - 6.5.2 Subdivision 61
  - 6.5.3 Don't Use Nice Names 61
  - 6.5.4 Disable Automatic Alignment 61
  - 6.5.5 Keep Alternates Next To Base Glyph 62
- 7 Spacing and Kerning 63**
  - 7.1 Spacing 63
    - 7.1.1 Spacing Shortcuts 63
    - 7.1.2 Linked Metrics 63
  - 7.2 Kerning 64
    - 7.2.1 Ways to Kern 64
    - 7.2.2 Kerning Groups 64
    - 7.2.3 Viewing Kerning Pairs 65
    - 7.2.4 Deleting Kerning Pairs 65
    - 7.2.5 Cleaning up Kerning 65
    - 7.2.6 Compressing Kerning 65
    - 7.2.7 Adding an Additional 'kern' Feature Lookup 66
  - 7.3 Sample Texts 66
    - 7.3.1 Placeholders 66
- 8 Hinting 67**
  - 8.1 Font-Wide Hinting Settings 67
    - 8.1.1 Standard Stems 67
    - 8.1.2 Alignment Zones 67
    - 8.1.3 Custom Parameters 69
  - 8.2 Autohinting 69
    - 8.2.1 Flex hints 69
  - 8.3 Manual hinting 70
    - 8.3.1 Stem Hints 70
    - 8.3.2 Ghost Hints 72
- 9 Multiple Master 73**
  - 9.1 Overview 73
  - 9.2 Set-Up 74
  - 9.3 Merging Two Files 74
  - 9.4 Fix Outline Incompatibility 74
  - 9.5 Layers Panel 76
  - 9.6 Ensuring Family Consistency across Files 76
- 10 Error Handling 77**
  - 10.1 Glyph Names 77
  - 10.2 Font Names 77
  - 10.3 Duplicate Unicode Values 78
  - 10.4 OpenType Feature Code 78
  - 10.5 Missing Outlines 80
    - 10.5.1 Open Paths 80
    - 10.5.2 Wrong Path Orientation 80
    - 10.5.3 Multiple Paths on Top of Each Other 80
    - 10.5.4 Outline Incompatibility 81
- 11 Import and Export 82**
  - 11.1 FontLab 82
    - 11.1.1 From FontLab to Glyphs 82
    - 11.1.2 From Glyphs to FontLab 82
  - 11.2 Robofont and Other UFO Tools 82
  - 11.3 OpenType and TrueType 82
    - 11.3.1 Opening Existing Fonts 82
    - 11.3.2 Generating Installable Fonts 83
- 12 Appendix 84**
  - 12.1 Automatic Feature Generation 84
  - 12.2 Custom Parameters 87
    - 12.2.1 Glyphs-Specific Parameters 87
    - 12.2.2 UFO3 Parameters 89

# 1 Glyphs



## 1.1 A TOOL FOR CREATING FONTS

Glyphs is primarily a tool for designing and producing new fonts. Its main principle is that you can edit glyphs in a word context. All tools are optimized for a type design workflow as natural, quick, and easy as possible.

We believe that you should be able to focus on your design and only be bothered with technicalities if it is really necessary. There is no need for keeping a design version next to a production version of your font. All production steps take place at export time. By default, Glyphs automates a lot of technical details for you, but you can always override automation manually.

Glyphs can open existing fonts. However, in the import process, some information stored in the font may be lost. For details, see chapter 11, 'Import and Export' (p. 82).

## 1.2 SYSTEM REQUIREMENTS

Glyphs is a Mac-only application. Glyphs 1.3 or higher requires at least OS X 10.6.6 Snow Leopard. If you have a PowerPC Mac running OS X 10.5 Leopard, you can still use Glyphs 1.2. You can download the latest PPC-enabled version from [glyphsapp.com/updates/latestPPC.php](http://glyphsapp.com/updates/latestPPC.php)

## 1.3 COMMUNITY

If you have questions or suggestions, you can register and post in the Glyphs forum at [glyphsapp.com/forums](http://glyphsapp.com/forums). Because of spam protection requirements, your first posting must be approved by one of the moderators.

## 1.4 UPDATES

The application is updated on a regular basis. You can check which version you have by choosing *Glyphs > About Glyphs* and trigger an update with *Glyphs > Check for Updates...* or by activating automatic updates in the application preferences. There, you can also activate Cutting Edge versions, which will provide you with the latest beta versions. Since these may contain experimental features and code re-writes, you are strongly advised to only work on copies of your files if you choose to activate the Cutting Edge option.

Because of contractual restrictions and the verification process, updates for the version purchased from Apple's Mac App Store can take significantly longer. For the same reasons, the App Store version cannot offer the Cutting Edge option.



## 1.5 GLYPHS MINI

Only available via the App Store, Glyphs Mini is a trimmed-down Light version of the application. It lacks many of the advanced features of the full application described in this handbook, e.g., support for plug-ins, layers, Multiple Master, Python scripting, custom OpenType feature code, custom parameters, and manual hinting. It is intended as a simple and affordable entry-level solution for casual type design or putting together a dingbat (symbol) font.

## 1.6 KEYBOARD SHORTCUTS

To enable a workflow as efficient as possible, Glyphs employs a number of keyboard shortcuts. You can set your own shortcuts in the Keyboard section of the System Preferences.

Some shortcuts conflict with default shortcuts for system functions like Spotlight or Spaces. To enable the full functionality of the application, it is advisable to change these system shortcuts as well.

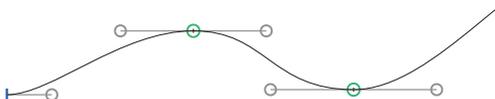
## 2 Edit View

### 2.1 DRAWING PATHS

#### 2.1.1 Freehand Outlines

Create paths with the Draw or Primitives tools. When the Draw tool  (shortcut P) is active, click to create straight lines, or click and drag to create curves. Move the on-curve point by holding down the space bar. Close the outline by clicking on its first node.

Nodes in smooth connections will appear as green circles. A smooth connection can either be a curve (an on-curve point in line with two surrounding off-curve points) or a tangent (an on-curve point in line with another on-curve point and an off-curve point). You can trigger the display of nodes with *View > Show Nodes*.



In order to draw a corner, hold down the Option key while dragging. Corner points are marked by blue circles. Points in a corner connection are not kept in line, so you can move them independently from each other.



Handles (Bézier control points, off-curve points) control the curvature of the path segment and are displayed as grey circles.

#### 2.1.2 Primitives

Glyphs offers rectangles and ovals as built-in primitive shapes. Click the tool or press F to activate it in its current

mode. Click and hold the Primitives tool or press Shift-F to choose between the two shape options.



Click once on the canvas to create a primitive by entering its measurements with the keyboard. Or click and drag to draw it directly into the edit area. Hold down Shift for a perfect square or circle. Hold down Option to draw from the center of the shape.

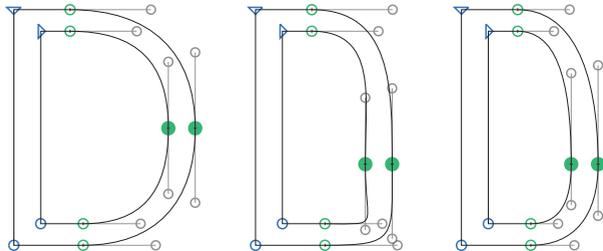
## 2.2 EDITING PATHS

### 2.2.1 Selecting, Moving, Converting

Click and drag with the Select tool  (shortcut V) to select nodes and handles inside a rectangular selection area. Hold down the Option key to ignore the handles and only select nodes. Double click on an outline segment to select complete paths. Hold down the Shift key to extend or reduce the selection. Glyphs allows you to select multiple handles independently of the nodes. Hold down the Option key to drag-select components.

Move the selection using the mouse or the cursor keys. Moving nodes will move the attached handles even if they are not selected. Hold down the Option key to move only explicitly selected on-curve points. While moving one or more nodes, hold down both Ctrl and Option (or add Option after you started dragging) to 'nudge' them, i.e., to proportionally adjust the surrounding handles at the same time.

Left: original glyph outline with two selected nodes.  
Center: selected nodes moved, handles stay the same.  
Right: selected nodes nudged, handles are adjusted.



Convert between smooth connections and corners by double clicking an on-curve point or selecting one or more nodes and pressing Return.

The *Layers > Tidy up Paths* command (Cmd-Opt-Shift-T) applies heuristics to set the appropriate mode for all nodes at once, or all selected nodes if there is a selection. It also removes superfluous points, e.g., handles on a straight segment.

Be careful when tidying up paths: In Multiple Master set-ups, superfluous points may be necessary for outline compatibility.

To move a handle, just drag it. Or select it and use your arrow keys. If more than one handle is selected, you can move them all simultaneously. When moving a single handle while holding down the Option key, its angle will be preserved.

**Tip:** Quickly add handles by Option-clicking on the outline between two nodes.

Option-clicking a line segment converts it into a curve segment, i.e., adds handles. To convert a curve back into a line segment, select and delete one or both handles.

### 2.2.2 Scaling and Rotating

The attributes of the selection are shown in the grey info box (*View > Show Info*):

**Tip:** In all number input fields throughout the application, you can use the up and down cursor keys to increase or decrease the value. Simultaneously holding down the Shift key gives you increments of 10.



When more than one node is selected, you can scale or move the selection by changing the numbers for its position (x and y) and its dimensions (w for width, h for height) in the info box. Set the transformation origin with the nine reference points on the left. Close the lock symbol to scale width and height proportionally. Open the lock to distort the selection, i.e., scale width and height independently from each other.

You can also rotate and scale your selection with the Rotate tool  (shortcut R) and the Scale tool  (shortcut S). Click to set the transformation origin, click and drag to rotate or scale, respectively. Hold down the Shift key to rotate in steps of ninety degrees and scale proportionally.

More path transformations are possible via the Palette. See the Palette chapter for further details.

### 2.2.3 Aligning

Choose *Layers > Align Selection* (Cmd-Shift-A) to quickly align all selected points. The command aligns both nodes and handles. Glyphs will then automatically choose between horizontal and

vertical alignment, whichever is smaller. The *Align Selection* command respects the transformation origin of the grey info box. Alternatively, you can either set the width (W) or the height (H) value of two or more selected points to zero in the same Info panel.

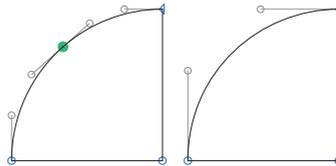
You can take advantage of the aligning behavior if you want to manage your serifs in components. For details, see: [glyphsapp.com/blog/serif-components/](http://glyphsapp.com/blog/serif-components/)

You can center an anchor horizontally between two points if you run the *Align Selection* command while two points and one anchor are selected. Running the command while one point and one component are selected will align the origin point (where the baseline crosses the left sidebearing) of the component to the selected node. The node keeps its position. If the component contains an anchor called 'origin', Glyphs will use that anchor instead of the origin point for aligning the component to a node.

To align complete paths and /or components to each other, use the Transformation section of the Palette (Cmd-Opt-P). See chapter 3, 'Palette' (p.34), for more details.

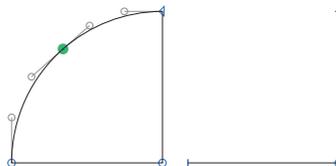
### 2.2.4 Deleting Nodes

Simply select a node and press the Delete key to delete the node. Alternatively, you can use the Erase tool  (shortcut E). Glyphs will keep the path closed and try to reconstruct the path segment without the node:



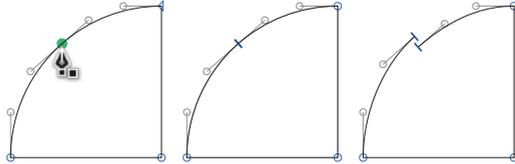
**Tip:** A quick way to get rid of a path segment between two nodes is to insert a point with the Draw tool (shortcut P) and immediately Option-delete it.

Hold down the Option key to break the path, i.e., remove the node and both path segments surrounding the node:



### 2.2.5 Opening and Closing Paths

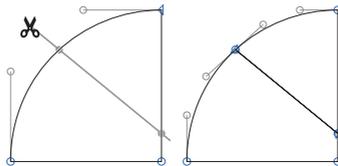
With the Draw tool  (shortcut P), click on a node to open the path in the position of the node. Open path endings are marked by perpendicular blue lines. You can now drag them apart with the Select tool  (shortcut V).



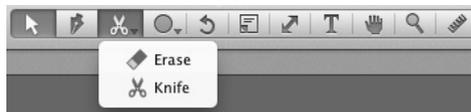
To close the path again, simply drag the open line endings onto each other using the Select tool.

### 2.2.6 Cutting Paths

With the Knife tool  (shortcut E or Shift-E), click and drag a line across a path, to cut the outline into two separate outlines. Glyphs will close the two resulting paths along the cutting line. If you cut across several overlapping paths, it will rewire the segments with each other.



To activate the Knife tool when it is not displayed in the toolbar, click and hold the Erase tool , and choose Knife from the pop-up menu. Alternatively, you can press Shift-E.

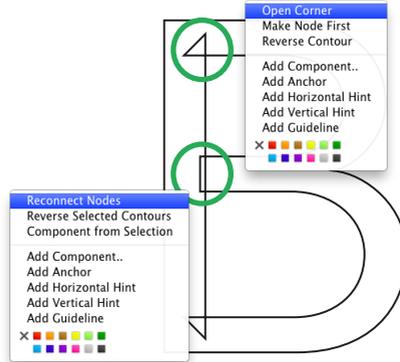


### 2.2.7 Resegmenting Outlines

In order to segment a closed path, select two nodes and select *Reconnect Nodes* from the context menu. To open the context

menu, right-click or Ctrl-click. If only one node is selected, the *Open Corner* command becomes available instead. It works in a similar fashion. Resegmenting your outlines this way allows to edit the path segments independently from each other. It also makes interpolating easier.

The results of the Open Corner and Reconnect Nodes operations. Opening corners only works on (blue) corner points.



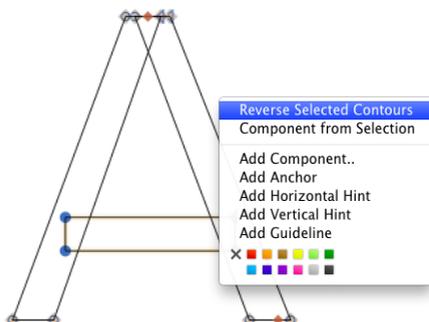
## 2.2.8 Controlling Path Direction

The starting point of a path is displayed as a triangle, indicating the path direction. Again, green and blue denote a smooth connection or a corner, respectively. You can make any on-curve node the first node by selecting it and picking *Make Node First* from the context menu.

**Tip:** The easiest and quickest way to get a hole in a glyph is to draw both inner and outer shapes, and press Cmd-Shift-R (Correct Path Direction).

All outer (black) paths need to run counter-clockwise, while (white) counters must go clockwise. You can change a path's direction by selecting it and choosing *Layers > Reverse Contours* or *Reverse Selected Contours* from the context menu. One node on each path will suffice as path selection. When no path is selected, you can use *Reverse Contours* from the *Layers* menu to toggle all path directions in a glyph. If you pick the *Correct Path Direction* command from the same menu, Glyphs performs an informed guess and finds the right path orientation for all contours in the selected glyphs. This will

also rearrange the contour order and reset the starting points of all paths to the top right nodes.



For interpolation, path order, starting points and path directions need to be compatible and consistent throughout all font masters. For more details on interpolation, see chapter 9, 'Multiple Master' (p. 73).

### 2.2.9 Extremes and Inflection Points

You can insert nodes on extremum points by Shift-clicking a segment with the Path tool (P). A node will be inserted at the nearest extremum or inflection point.

Alternatively, you can also choose *Layers > Add Extremes* and nodes will be added at extremes on all paths of the active layer. Glyphs will not add an extreme if the resulting segment would be very short. In this case, it assumes that the node placement was intentional.

## 2.3 GUIDELINES

### 2.3.1 Magnetic Guidelines

If you drag nodes across the canvas, red lines will appear, indicating when you are aligned with other nodes or a vertical metric. You can temporarily deactivate magnetic guidelines by holding down the Ctrl key.

### 2.3.2 Local and Global Guidelines

To add a guideline, right- or Ctrl-click to open the context menu and choose *Add Guideline*. A horizontal blue line appears. You can activate it by clicking on it, alter its position by clicking and dragging the little circle. Double click the circle

to turn it perpendicular to its original orientation. Click and drag a guideline to rotate it.

To turn it into a global guideline, i.e., a guideline that appears in all glyphs throughout the master, right- or Ctrl-click the circle and choose Make Global Guideline from the context menu. The guideline will turn red. To make it local again, cut and paste it. Pasted guidelines are always local.



When a guideline is selected, you can also enter values for its position and its angle in the grey info box. By default, a guideline will be set relative to the left sidebearing. However, if you change the alignment of the guideline by clicking on the Alignment icon **▶** in the info box, the guideline will stay fixed relative to the right sidebearing. This is useful for right-to-left scripts, slanted global guidelines or when the right sidebearing is changed.

With the Measurement checkbox, you can turn it into a measurement guideline. For more details on measurement guidelines, see the section 2.7, ‘Measuring’ (p.23).

## 2.4 GLYPH DISPLAY

### 2.4.1 Zooming

There are many ways to zoom in and out in Edit view. If you are on a MacBook or have a trackpad to your disposal, you can use pinch and stretch gestures. Or hold down the Option key and use a scroll gesture or the scroll wheel of your mouse. Or activate the Zoom tool  (shortcut Z) and click in the canvas to zoom in, Opt-click to zoom out, or click and drag across an area that will be zoomed to fill the window. You can temporarily activate the Zoom tool by holding down Cmd-Space for zooming in, or Cmd-Opt-Space for zooming out.

Or you can use the zoom commands from the *View* menu: *Zoom In* (Cmd-plus) and *Zoom Out* (Cmd-dash). *Zoom to Active Layer* (Cmd-zero) will maximize the area between ascender

**Tip:** You may need to change your Spotlight shortcut in the System Preferences for the Cmd-Space shortcut to work in Glyphs.

and descender in the window. *Zoom to Actual Size* (Cmd-Opt-zero) will zoom one font unit to the size of one screen pixel.



Or you can use the zoom buttons in the bottom right corner of the window. Alternatively, you can set the zoom value numerically by entering a point height in the field between the buttons. The value specifies at which size 1000 units will be displayed. Since Mac OS X assumes a screen resolution of 72 ppi, one point corresponds to one actual screen pixel, so if you enter a value of 1000, one unit will zoom to one pixel.

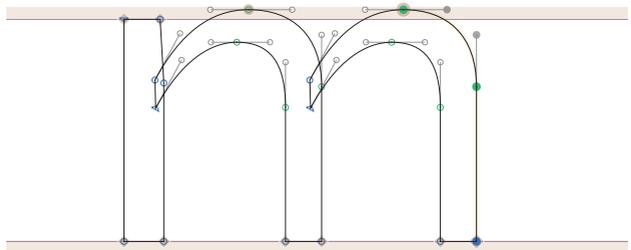
### 2.4.2 Panning

You can use trackpad panning gestures. Use the wheel of your mouse to scroll vertically, hold down shift to scroll horizontally.

**Tip:** Cmd-Space may interfere with the system-wide shortcut for invoking Spotlight. You can change it in the System Preferences.

Alternatively, you can switch to the Hand tool  (shortcut H) and drag the canvas around, or simply hold down the space bar to temporarily switch to the Hand tool. If you are in text mode, simply pressing the space bar would add a word space to your sample text. To avoid that, press Cmd-Space and subsequently release the command key.

### 2.4.3 Nodes in Alignment Zones



Nodes located exactly on a vertical metric line (see the Font Info chapter for further details) are highlighted with a beige diamond. Inside an alignment zone (see 6.2, ‘Masters’, p. 55), the highlighting assumes the shape of a circle. This helps controlling the position of nodes even at small zoom scales.

#### 2.4.4 Components, Anchors, Clouds

Letters built from components, so-called compound or composite glyphs, are handled in a special way. Accented or marked letters, built from a base letter and a mark, automatically inherit the spacing from the base glyph. The mark is positioned according to the anchor positions. The accented letters are always in sync with the base letter, even if spacing or anchor positions are changed. Figures and other non-letters form an exception to this rule. Such compound glyphs are not in sync with their base glyphs. This, for instance, allows to easily re-use proportional figures for tabular figures or vice versa.

You can turn all components inside a glyph into editable paths by choosing *Layers > Decompose Components*. Triggering *Decompose* from the context menu of a specific component only decomposes the selected component.

Automatic alignment of components is disabled as soon as there are any paths in the glyph. Once those paths are removed again, the compound glyph snaps back into automatic alignment. To explicitly disable automatic alignment in pure compound glyphs, right-click on a component and choose 'Disable Automatic Alignment' from the context menu. For a font-wide deactivation, choose the option of the same name in *File > Font Info > Other Settings*.

To edit the base glyph of a component, double click the component and Glyphs will place the original letter next to the compound glyph. Alternatively, you can select the component inside the composite glyph and click the arrow symbol that appears in the grey info box (Cmd-Shift-I). Placeholders can indicate an 'Empty Base Glyph' and a 'Missing Base Glyph'. Double clicking the placeholder will open an empty glyph or create and open a missing glyph, respectively.



Anchors are added to most glyphs automatically with the *Set Anchors* command from the *Layers* menu (Cmd-U). If you additionally hold down the Option key, Glyphs will delete all anchors in the selected letters and reset the standard anchors.

Setting and resetting anchors also works for multiple glyphs at once. Glyphs will respect the italic angle of the master.

Once you have anchors in your letters, all you need to do is refine their position. An anchor in the base glyph is connected with the anchor in the mark glyph by sharing the same name and additional underscore prefix, e.g., a 'top' anchor in the base glyph and a '\_top' anchor in the mark are used to place the mark glyph in the compound glyphs. Select an anchor and two points and choose *Align Selection* (Cmd-Shift-A) from the *Layers* menu to horizontally center the anchor between the points. Again, Glyphs will respect the italic angle of the master.

**Pro User Tip:** If you want to control which marks are associated with which base glyph, create a copy of `GlyphData.xml` in the Info folder you can find via *Script > Open Scripts Folder*. Create it next to the *Scripts* folder if it is not there yet. Then, edit the anchors and accents attributes.

You will find the XML file in the Package Contents of the Glyphs application. Drill down to `Contents/Frameworks/GlyphsCore.framework/Versions/A/Resources`.

Build compound glyphs by either creating a glyph with the proper name, e.g., 'aacute', or by selecting one or more existing glyphs and choosing *Layers > Make Component Glyph* (Cmd-Opt-Shift-C) to rebuild the glyph as a composite. Glyphs will use a built-in database of glyph compositions to determine the components of the glyph in question. You can add components yourself to an existing glyph by choosing *Layers > Add Component ...* (Cmd-Shift-C) and picking the base glyph.

If you have already drawn a letter that you want to turn into a component-based glyph, you can force the composition with *Layers > Make Component Glyph* if the individual components already exist as separate glyphs in the font. If they do not already exist, you can create a new component from an existing path in an outline layer by selecting the path you want to turn into a component and choosing *Layers > Component from Selection* or picking *Component from Selection* from the context menu. Glyphs will then suggest a name of the component based on the decomposition information from the built-in glyph database. Once you confirm the dialog, Glyphs will create a new glyph containing the the selected path and place it as a component in the original glyph. This mechanism is useful for deriving dotaccent and dotlessi from i, for instance.

When creating new compound glyphs, you can use composition recipes and force a non-standard composition. You can build recipes in three ways:

- `A=a` *base* → *copy*
- `x+dieresis=xdieresis` *base + mark* → *diacritic*
- `s.sc+s.sc=germandbls.sc` *base + base* → *ligature*

In the first example, the copy recipe, Glyphs will create a glyph called 'a' with 'A' placed in it as a component. In the

second example, the diacritic recipe, a glyph called ‘xdieresis’ will be generated with ‘x’ as the base and ‘dieresis’ as the diacritic mark. If available, anchors will be used for the mark placement. In the last example, the ligature recipe, Glyphs will create a glyph called ‘germandbls.sc’ with two components ‘s.sc’ next to each other.

Glyphs allows you to nest components. For example, you can build the ‘dieresis’ glyph out of two ‘dotaccent’ components. Subsequently, you can use this compound ‘dieresis’ in higher-level compounds such as ‘adieresis’ (ä).

Select an anchor in the base glyph to get a preview of the most common accents that may attach to this glyph. Similarly, select an anchor in the mark glyph and this accent is shown on all other glyphs in the same Edit view.

You can add anchors by right-clicking and choosing *Add Anchor* from the context menu. You will immediately be prompted for a name. Enter an ASCII-only name without whitespace and confirm by pressing Return. You can change the name of an existing anchor by either clicking on it and pressing Return or double clicking it, then you can start typing the new name.



Sometimes you need more than one ‘top’ anchor, for example in Arabic ligatures or Vietnamese letters. In this case, add an underscore and a number or a suffix to the anchor name, e.g., ‘top\_viet’ or ‘top\_1’, ‘top\_2’ and so on. Then, in the compound glyph, select the mark, and via the anchor symbol in the grey info box, select to which anchor the mark component should attach. The anchor symbol becomes visible if there is more than one anchor that fits.



When building letters from several base glyphs, the components are set next to each other with the respective spacing and kerning for this combination. For example when building a fraction out of one.numr, fraction, and two.dnom,

**Tip:** Quickly disable automatic alignment for all components in a glyph by simply adding a node with the Path tool (P). A single node already constitutes an open path, and as such, it will be ignored at export time.

the fraction will look as if its parts were typed individually. To move them to a different position, you may need to disable the automatic alignment of a component via the context menu. Components can always be moved freely if there is at least one path in the glyph.

#### **2.4.5 Special Marks for Glyph Composition**

When building compound glyphs, Glyphs will prefer marks that carry the same name suffix. For instance, when composing ‘adieresis.sc’, Glyphs will prefer ‘dieresis.sc’ to ‘dieresis’. For uppercase letters, marks with a ‘case’ endings are preferred.

When building combinations for i and j, make sure both ‘dotlessi’ and ‘dotlessj’ are in your font. This is because the dot is typically not included when i and j have accent marks. For i and j diacritics, Glyphs will prefer marks carrying a ‘i’ or ‘narrow’ suffix.

#### **2.4.6 Mark to Base Positioning**

Glyphs can automatically build the ‘mark’ (Mark to Base) feature from combining (non-spacing) diacritical marks containing underscore anchors, e.g., ‘\_bottom’ or ‘\_top’, in combination with all base glyphs that carry corresponding regular anchors, e.g., ‘bottom’ or ‘top’. Latin combining diacritical marks usually carry a ‘comb’ in their names, e.g., ‘acutecomb’ or ‘macroncomb’.

Combining diacritical marks have their own Unicodes and thus can be typed or inserted in a text. This way, a user can place any mark on any base letter, by first typing the regular letter, and then inserting the combining mark.

Alongside the ‘mark’ feature, Glyphs will also build the ‘ccmp’ (Glyph Composition and Decomposition) if glyphs like dotlessi and dotlessj are present.

#### **2.4.7 Mark to Mark Positioning**

If both underscore and regular anchors are present in a combining diacritical mark, Glyphs will also automatically build the ‘mkmk’ (Mark to Mark) feature. A user will then be able to stack any combining mark on any other combining mark carrying both anchors.

### 2.4.8 Cursive Attachment Anchors

To enable true cursive attachment in Arabic typesetting, add anchors called 'exit' and 'entry' to the respective stroke endings and beginnings in medial, final, and initial letter forms. The entry anchor of the instroke will be connected to the exit anchor of the preceding outstroke. You can preview cursive attachment immediately in the Edit view when right-to-left typesetting is enabled (see 2.6.3, 'Writing Direction', p. 23).

## 2.5 BACKGROUND

The background layer is useful for storing a path temporarily. Filters may use the background as backup layer in order to work non-destructively. To activate the background, choose *Edit Background* from the *Layers* menu (Cmd-B). The window display will darken slightly to indicate that you are in the background layer.

*Layers > Selection to Background* (Cmd-K) replaces the current content of the background with the active selection; this works in reverse when the background is active. Simultaneously holding down the Option key (or pressing Cmd-Opt-K) adds the current selection to the background without clearing it first.

Via *Layer > Assign Background*, you can put another font in the background layer of all selected glyphs. You can also put the same font into its own background in order to keep track of outline manipulations.

## 2.6 ENTERING TEXT

The Edit view also acts as a simple text editor, and as such allows you to edit your glyphs in the context of a whole word or even sentence. As long as the Text tool (T) is active, it accepts input via the current input source set in the System Preferences, the Keyboard Viewer, the Character Viewer, but also via the clipboard or *Edit > Special Characters*. In OS X 10.8 Mountain Lion, you can even use the speech recognition features and dictate your text.

The Edit view is limited to a certain width. You can set the maximum width in the application preferences. Select

**Tip:** To quickly switch back from text entry to editing the current glyph, press the Esc key..

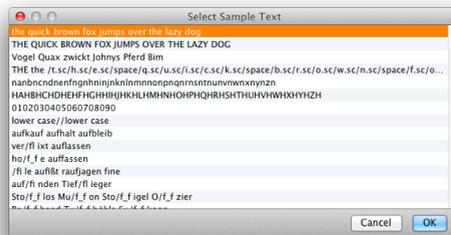
*Glyphs > Preferences > User Settings* and edit the Text View Width value accordingly.



### 2.6.1 Sample Texts

You can edit and store any number of sample texts in the application preferences. Select *Glyphs > Preferences > Sample Strings*. You can also enter individual glyph names if you escape them with a leading slash and a trailing space, e.g., 'H/ adieresis.sso1 mmer' will result in 'Hämmer' with the first Stylistic Set variation of 'ä'.

To make an Edit tab display any of these texts, choose *Edit > Select Sample Text* (Cmd-Opt-F). In the appearing dialog, use your arrow keys or click on a line to pick the sample text and press Return or click the OK button afterwards.



### 2.6.2 Text Tool

Select the Text tool (shortcut T) to switch to text mode and start typing. You can enter multiple words, complete

sentences, even line breaks. You can copy and paste text into and from the Edit tab. The arrow keys, the Edit commands and OS X Application Services work as they would in any Mac app.

The current glyph is the one to the right of the cursor. You can switch to the previous or next glyph in the font by pressing the Home and End key, respectively. Add Shift to advance through the glyphs as they are currently visible in the *Font* tab. This is useful when you filter glyphs in the Font view and then want to step through each one of them.

*Edit > Add Placeholder* (Cmd-Opt-Shift-P) inserts a placeholder for the current glyph. Placeholders are dynamically replaced by the currently selected glyph.

To insert a glyph that you cannot or do not know how to type with the keyboard, choose Find from the *Edit > Find* submenu (Cmd-F). In the dialog that appears, enter the name or a part of the name of the glyph. The dialog will show a list of glyphs whose name contains the text you entered. Select the glyph you want and press Return or click the Select button to insert it into your sample text.

**Tip:** On a MacBook, you can simulate Home and End by pressing Fn-leftarrow and Fn-rightarrow..



### 2.6.3 Writing Direction

In Edit view, switch between left-to-right, right-to-left, and top-down with the respective alignment buttons in the bottom right corner of the application window. The 'To' button next to it toggles kerning.



## 2.7 MEASURING

Glyphs offers several ways to determine coordinates and measure distances between points and curves.

### 2.7.1 Info Box

You can toggle the display of the grey info box via the *Show Info* command in the *View* menu (Cmd-Shift-I). The info box always displays data relevant to the current selection. If there is exactly one node selected, its coordinates will be displayed.



Select a handle (off-curve point, Bézier control point), and the info box will also carry its delta values (x and y difference

to the on-curve point) and the total length of the handle (distance to the on-curve point).



x: 387 y: 68 Δx: -29 Δy: -17 l: 33.62

**Tip:** To quickly and precisely measure a stem or bowl width, select two nodes that indicate the width and see what the info box displays next to 'w':

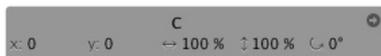
If you select more than one point, be it on- or off-curve points, the info box will display the width (w) and the height (h) of the box defined by the current selection. The x and y coordinates displayed describe the position of the bounding box of the selection as indicated by the grid, i.e., the bottom center point of the grid indicates that the coordinates describe the bottom center point of the bounding box.



x: 160 y: 600 2 Nodes Selected  
w: 78 h: 0

In the info box, any displayed number can be selected and changed by entering a new value. You can use the Tab key to advance to the next number slot displayed. Changes take effect once you press Return or tab out of a value. If you scale a selection this way, the lock symbol toggles proportional x/y scaling, and the grid controls the scaling origin.

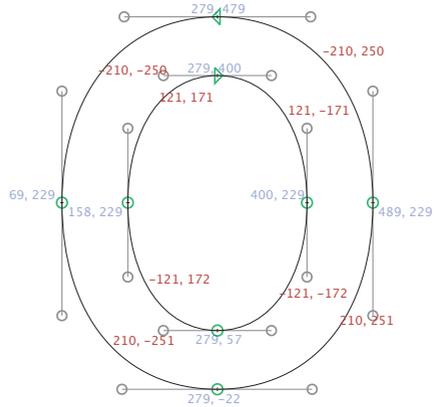
When a component is selected, the info box will display the name of the original glyph the component points to, its x and y offset, its horizontal and vertical scale in percent, and its counter-clockwise rotation angle. The little arrow button in the top right corner will insert the original glyph to the left of the current glyph, and activate it for editing. For more details on working with components, see section 2.4.4, 'Components, Anchors, Clouds' (p. 17).



C  
x: 0 y: 0 ↔ 100% ↑ 100% ↻ 0°

### 2.7.2 Measurement Tool

Switching to the Measurement Tool  (shortcut L) allows you to see all coordinates at once.



The x delta values respect the italic angle set in Font Info. So, an x delta of zero indicates a line exactly in the italic angle.

The blue numbers are the x and y coordinates of the on-curve points, the red numbers are the x and y delta values between the on-curve points. These values help you check the symmetry of your curves.

Clicking and dragging draws a ruler that displays precise measures between all of its intersections with outlines. At the end of the ruler, its angle is displayed in counter-clockwise degrees, where zero degrees corresponds to dragging the ruler completely horizontally to the right.

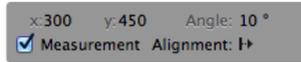


You can temporarily activate the ruler and the display of point coordinates by simultaneously holding down Ctrl, Option and Command.

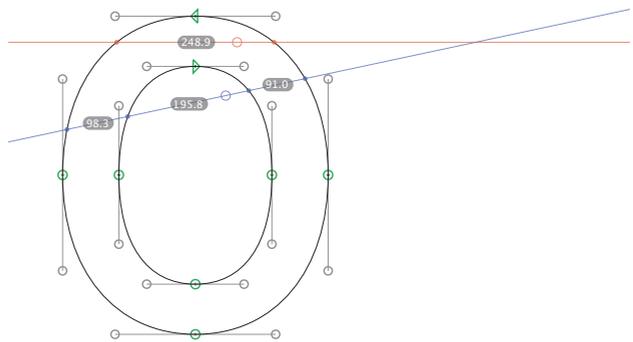
### 2.7.3 Measurement Guidelines

Any guideline, even a global guideline, can be turned into a measurement guideline. Simply select the guideline by clicking on its selection handle and activate the Measurement check box in the grey info box. You can set the angle of the

measurement guideline by selecting it and either grabbing it anywhere besides its selection handle or setting the angle numerically in the info box.



Similar to rulers, the measurement lines will display the distance between their intersections with outlines. Contrary to rulers, they do so as long as guidelines are shown, no matter which tool is active. Like regular guidelines, measurement lines can be both local (blue) and global (red).

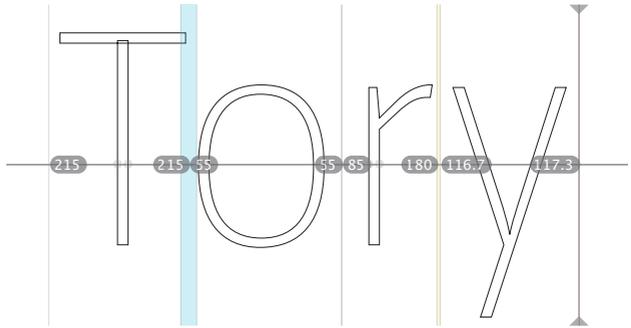


### 2.7.4 Measurement Line

When you are in text mode, you can enter a measurement mode by choosing *Show Measurement Line* from the *View* menu (Ctrl-Command-Opt-M). The measurement line will display the sidebearings at a given height, ignoring the shape of the glyph at other positions. You can alter its height by Ctrl-Command-Opt-clicking or Ctrl-Command-Opt-dragging. Or you can switch to the Measurement tool and simply drag it.

Thin grey lines indicate the widths of the glyphs. The numbers displayed indicate the distance between the left or right sidebearing and the point where the measurement line first crosses the glyph outline. Kernings receive a color

code. Negative kernings will be displayed light blue, positive kernings yellow.



## 2.8 ANNOTATING

The Annotation tool  (shortcut A) allows you to add simple notes and correction marks to your drawings. Once you activate the tool, the grey info box (*View > Show Info*) will turn into a little palette holding a range of annotation tools.

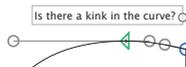


### 2.8.1 Annotation Cursor

The first tool on the Annotation palette is a simple edit tool for annotation marks. It allows you to activate, move, and resize existing annotations. Once an annotation mark is activated, you can delete it by pressing the Delete key.

### 2.8.2 Annotation Text

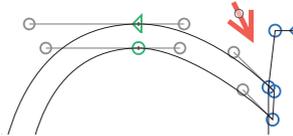
The second tool from the left is a simple text tool. Activate it by clicking on the T button and then click anywhere on the canvas to add a text box. Double click on the text and start typing. When you're finished, just activate the next tool you want to work with. You do not need to acknowledge the text entry.



The handle on the right controls the width of the text box. The height of the box always automatically adjusts to the length of the entered text.

### 2.8.3 Annotation Arrow

The third tool on the Annotation palette allows you to put red arrows on the canvas. The handle on the arrow stem controls the rotation of the arrow.



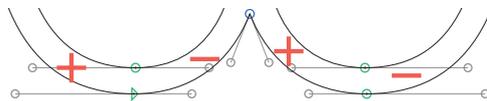
### 2.8.4 Annotation Circle

The fourth tool puts red highlighting circles on the canvas. The handle at the bottom of the circle controls the diameter.



### 2.8.5 Plus and Minus Annotations

Many designers use plus and minus signs to indicate that a counter, a bowl or a stem needs to be thickened or thinned, respectively. Click on the plus or minus button and then on the canvas to add the symbols to the editing area.



## 2.9 IMAGES

### 2.9.1 Adding Images

You can add all image files supported by OS X, including PDFs, to any glyph layer simply by dragging them into the Edit area or by choosing *Layers > Add Image ...* (Cmd-Opt-Shift-I). You can toggle the display of images via *View > Show Image*. By default, images will be scaled to a size where one DTP point corresponds to one font unit, and placed at the origin point of the layer.

**Tip:** You can add many images at once in Font View. See the Font View chapter for details.

## 2.9.2 Manipulating Images

Move the image by dragging it to a new position. When an image is selected, the grey info box (Cmd-Shift-I) allows you to numerically control position (x, y), horizontal (h) and vertical scale (v), as well as cropping from the top (t), the bottom (b), the left edge (l), and the right edge (r) of the image. The arrow symbol will reveal the original image file in the Finder. A click on the lock symbol freezes the image status until you unlock it again via the context menu.



Besides locking and unlocking an image, the context menu lets you crop the image to the layer bounds (width, descender, and ascender) and reveal the image file in Finder.

In the Glyphs document, only the relative path of the image is stored. Thus it is a good idea to keep them in a subfolder next to the Glyphs file. Image files will be ignored at OTF export, but can be exported in a PhotoFont.

## 2.10 PREVIEWING AND TESTING

### 2.10.1 Previewing Kerning

Kerning Preview is on by default. You can activate and deactivate kerning with the Preview Kerning button in the bottom right corner of the window.



### 2.10.2 Previewing Masters

The Edit view already is a preview of the master(s). It always previews the anti-aliased outlines and the kerning of the currently selected font master. Switch between masters with the master button at the top of the window or by pressing Cmd and the number of the master, e.g., Cmd-1 and Cmd-2.



### 2.10.3 Previewing Path Offset

Via *View > Show Preview Offset*, you can toggle a grey preview of the offset the *Offset Paths* filter would produce with its current settings. This is useful for an open paths workflow. For the preview to take effect, *Filter > Offset Paths* must have been run at least once. The preview only applies to the current glyph.

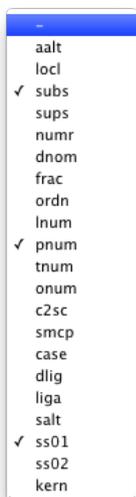
A lowercase script p with Preview Offset enabled (left) and disabled (right).

The offset value is taken from the settings used the last time the *Offset Paths* filter has been run.



### 2.10.4 Previewing OpenType Features

You can activate and deactivate features for previewing through the Features menu at the bottom left of the window in Edit view:



You may need to recompile the features before they are available in the pop-up list. You can select any number of features concurrently, and deselect all of them at once by choosing the dash at the top of the menu.

In Edit view, glyphs can only show a preview of substitution features, since positioning features can be handled very differently in different application and system environments. To test positioning features, it is therefore recommended to make use of the Adobe Fonts folder. For details, please see section 2.10.7, 'Previewing in Adobe applications' (p. 32).

### 2.10.5 Previewing Interpolated Instances

Click on the Preview button  next to the Features pop-up. The window content will be sectioned vertically, the Edit view at the top, the Preview at the bottom. Pick an instance from the instances pop-up next to the Preview button to see a live interpolation of the respective instance in the Preview area. Drag the separator line to adjust the size of

the Preview. The Preview gives you the complete text on a single line and centers on the current glyph. You can double click a letter in the Preview to make the Edit view center on it. The Preview respects custom parameters as well as the bracket trick described in the section 3.4.2, 'Switching Glyph Shapes' (p. 36).

You can switch between black-on-white and white-on-black with the Invert button  next to the Instances. To test the legibility of your design, you can use the slider to blur the font sample in the Preview.



## 2.10.6 Previewing in OS X

Because of the complicated font caching mechanism employed by OS X, simply overwriting a previously installed font can lead to a range of problems. In order to avoid such font cache difficulties, you can change the family name in the Font Info every time you export, e.g., by adding an increasing number ('MyFont 1', 'MyFont 2', 'MyFont 3' etc.) or a letter combination ('MyFont AA', 'MyFont AB' etc.). The latter scheme ensures a more consistent font menu ordering if you have many versions of your font installed.

If you do run into cache problems, close all running applications and remove all instances of your font from Font Book or the Fonts folder. Then, open the Terminal application and type these lines, each of them followed by a return. The second command will prompt you for your administrator password:

```
atsutil databases -removeUser  
sudo atsutil databases -remove  
atsutil server -shutdown  
atsutil server -ping
```

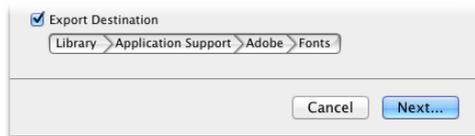
After these Terminal commands, you need to restart the Mac as quickly as possible. If the problems persist, boot the Mac in safe mode and recreate caches by holding down the Shift key while restarting and logging in.

Because of the described difficulties with the OS X font cache, we do not recommend testing unfinished fonts in Font Book.

### 2.10.7 Previewing in Adobe applications

For a complete preview including things like positioning features and menu ordering, just pick *File > Export* (Cmd-E) and choose */Library/Application Support/Adobe/Fonts/* as Export Destination. The font becomes immediately available in all Adobe applications. Any previously saved instance of the font in this folder will be overwritten. The font will not be available outside Adobe apps, but this is a convenient way to circumvent any font cache problems in OS X.

If the Fonts folder does not exist, you can create it right in the Open Folder dialog by pressing Cmd-Shift-N. In this case, you have to restart any running Adobe apps for the change to take effect.



## 2.11 EXCHANGING OUTLINES WITH ILLUSTRATOR

### 2.11.1 Preparation

First, find the right scale for your drawings in Adobe Illustrator. One point in Illustrator corresponds to one unit in Glyphs, i.e., an element that is 100 pt high will end up at a height of 100 units in Glyphs. To quickly get the right scale, draw a rectangle with the height of the capital letters in Glyphs, copy and paste it into an Illustrator artboard and scale the drawings to fit the height of the rectangle.

Alternatively, you can start in Illustrator and set the artboard to a height of your UPM size in points, e.g., 1000 pt if your UPM is 1000. Both ascender and descender should fit inside the artboard.

### 2.11.2 Set the Page Origin

If you set the page origin to the intersection of base line and LSB, the paths will have the right position.

In Illustrator up to Version CS4, set the origin of the page by dragging the cross hair in the top left corner between the rulers. In Illustrator CS5 or later, set the origin of the page in the Artboard Options.

### **2.11.3 Copy the Paths**

Copy and paste the outlines. A dialog may appear, preventing you from pasting something far outside the bounds of the letter. It will move the path next to the origin, i.e., the intersection of base line and left sidebearing.

## 3 Palette

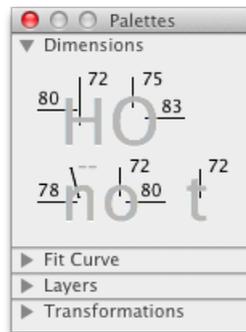
### 3.1 PALETTE WINDOW

You can open the floating Palette window from the *Window* menu, or press the shortcut, *Cmd-Opt-Shift-P*. The Palette has four sections. You can collapse or expand them by clicking on their title or the triangle next to it. If you abut the Palette against the left or right edge of your screen, the main window zoom (click the green zoom button or choose *Window > Zoom*) will save space for the Palette.

Most Palette transformations work in both the Font and the Edit view. Depending on the selection, the transformations work on paths, parts of paths, or complete glyph layers, even when multiple glyphs are selected.

### 3.2 DIMENSIONS

The entries in the Dimensions section have no effect on any font parameters. They serve as a cheat sheet for your design process. You can enter values for a couple of crucial measures in your current layer. Values are stored per master.



### 3.3 FIT CURVE

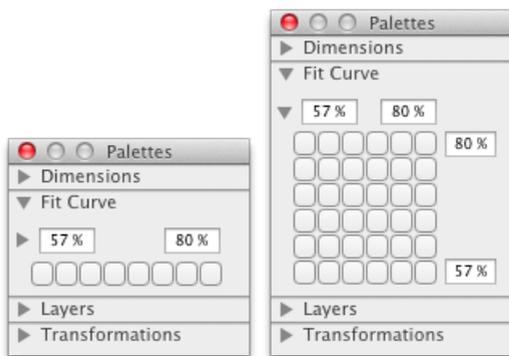
The Fit Curve panel helps creating curves with matching curvatures. In most instances, the collapsed (i.e., single line) view will do. Select percentages by setting the left field to the minimum value and the right field to the maximum value. The buttons in between interpolate these two values. Alternatively, you can press *Ctrl-Opt-1* through *8*. For the Fit Curve function to take effect, at least one handle must

be selected. Fit Curve also works when handles of different segments are selected.

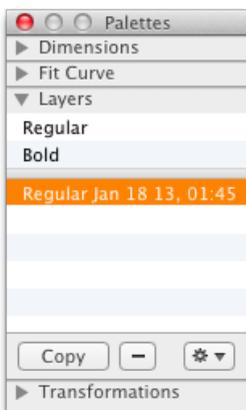
The numbers describe the percentage length of the handle. The distance from the curve point to the intersection of the handles equals 100 percent. 57 percent give you an elliptic curvature.

For finer tuning, click the little triangle to expand the Fit Curve panel into two dimensions. In that case, the first handle on each curve segment is controlled by the x axis, while the second one is controlled by the y axis.

Smaller curves, e.g. counters, generally need higher curvature percentages than larger curves, e.g. on the outside of a bowl.



Click one of the little buttons to alter the curvature of the selected curve segments. Click the same button to adjust other curve segments (e.g., in other letters of the same font) accordingly. This way, you can achieve matching curvatures in different curves.



### 3.4 LAYERS

Glyphs differentiates between two sorts of layers: font masters and regular layers. Font masters are needed for interpolating instances. All glyphs in a font will always carry all font masters set in *File > Font Info > Masters* (see 9, 'Multiple Master', p.73). In the Layers section of the Palette, master layers are displayed above a separator line.

Regular layers are displayed below the separator. They are stored on a per-glyph basis. You can have any number of layers in a glyph. Use them for keeping variations of the respective glyph.

### 3.4.1 Working with Layers

Font masters cannot be edited through the Palette. They are controlled via the Font Info window.

Select an available master from above the separator and click the Copy button to create a new layer. A copy of the master will appear below the separator line, carrying the name of the master and the creation date. Double click the layer name to edit it. To delete a layer, select it and click the minus button below. To exchange it with the glyph layer that is used as an interpolation master, click the gear button and select Use as Master from the menu that pops up.

### 3.4.2 Switching Glyph Shapes

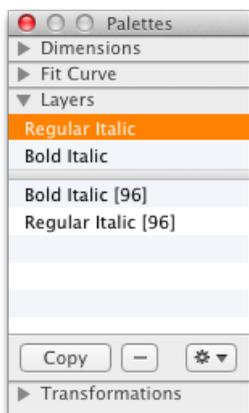
In a Multiple Master font with a weight axis and no other axes, you can let Glyphs exchange layers with masters automatically for a range of weights. For this to happen, create layer copies of the masters you want to exchange. Set each layer name to the name of the corresponding master, followed by a space and a number in brackets, e.g., 'Light [96]' or 'Regular [210]'. For instances with weight values above the bracketed number, Glyphs will use these layers as masters.

If the master shapes are not point-compatible with the layer copies, you need to exchange both masters at the same interpolation stage. In that case, the names of both layer copies should carry the same weight value in brackets. Otherwise an instance in between would end up with incompatible masters.

A common usage example is the vertical bar that crosses down through the dollar sign. In bolder variations of the glyph, most designers choose to split it in two and reduce it to whatever extends below and above the s-shaped part of the sign. Keep the light variations as masters and put the bold variants into the bracketed layers.



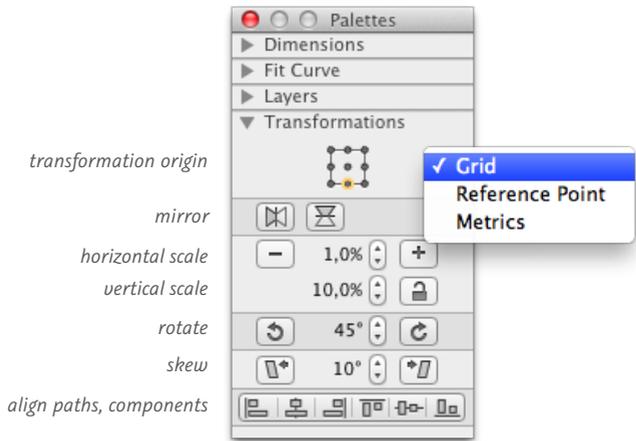
This 'bracket trick' only works in single axis set-ups. If you want to switch glyph shapes in multiple axis set-ups, you can use separate glyphs and rename them at export with



custom parameters in the respective instances. See section 6.3, 'Instances' (p.57), for more details.

### 3.5 TRANSFORMATIONS

The bottom section of the palette is reserved for basic path transformations. First, you need to select a transformation origin. This can either be a Grid point calculated relative to the selected paths, a manually set Reference Point, or one of the Metrics taken from the entries in the Masters section of the Font Info, such as baseline, (half or full) x-height and (half or full) cap height.



Once you have set the transformation origin, you can, from top to bottom:

- mirror the selection horizontally or vertically
- scale the selection by percentage values
- rotate the selection counterclockwise or clockwise by degrees
- skew the selection left or right by degrees
- align complete paths or components to each other

The scaling takes two percentage values, for horizontal and vertical scaling respectively. When the lock symbol is closed, the second value will be ignored and the first value applies to both x and y scaling. The entered values refer to positive scaling which is activated by pushing the plus button. The minus button does not scale down by the entered values, but

This means that if you want to scale down all coordinates by half, you need to enter 100 percent and hit the minus button.

reverses the positive transformation. This way, you can always undo a positive scale with the minus button and vice versa.

The align buttons only work on complete paths, no matter whether they are partially or fully selected. If you want to align individual points or segments, you need to use the *Align Selection* command from the *Layers* menu. See section 2.2, 'Editing Paths' (p. 9), for more details.

## 4 Filters

### 4.1 FILTERS MENU

You can apply filters to the active layer in the Edit view, or to any number of selected layers (with the text tool), or in the Font view. Filters only affect visible layers, never all layers in a glyph. The following filters come standard with Glyphs.

### 4.2 HATCH OUTLINE

Creates hatched letters. Distance, width, and angle of the hatch-lines can be chosen. Glyphs always uses the background path as source. If there is no path, Glyphs will put a copy of the current path into the background. Alter the appearance of the hatching by changing the background path and applying the filter once again.

### 4.3 OFFSET CURVE

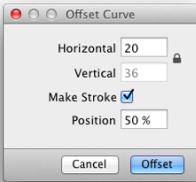
Changes the thickness of stems horizontally and / or vertically. The lock sign uses the horizontal value for both horizontal and vertical expansion. Without the Make Stroke option, paths will just be moved into a parallel position:



With the Make Stroke option, selected paths will be expanded to closed outlines:



The Position setting controls the distribution of the expansion. At 0%, the path will only expand to the right. At 100%, the path will only expand to the left. At 50%, the expansion will be evenly distributed to both sides of the path. Right and left sides are determined by the path orientation.



**Tip:** You can apply the Make Stroke option to an open path to get a thickened outline. This way, you can get an initial outline just by drawing the internal stroke.

## 4.4 REMOVE OVERLAP

Removes overlaps of selected paths as well as open paths and stray nodes. If no paths are selected, the filter will be applied to all paths of the selected letter(s). The filter expects all outline orientations to be set correctly (see 2.2.8, 'Controlling Path Direction', p. 13). To maintain base shapes while editing, you can have overlaps removed automatically at export with the respective option in the export dialog (see 11.3.2, 'Generating Installable Fonts', p. 83).

## 4.5 ROUGHEN

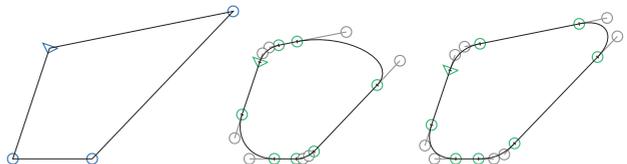
Segments an outline into straight subpaths and randomly moves the nodes within a given limit. Control the size of the subpaths with the Segment Length field. The Horizontal and Vertical values control the maximum offset for each node. With the Angle value, you can tilt the horizontal and vertical node transformation. If you use modest values, the resulting glyphs will receive a roughened look, hence the name:



## 4.6 ROUND CORNERS

Use this filter to round all selected corners of a path. To only round the outside corners, i.e., corners pointing into the white background, simply select nothing. Use the *Visual corrections* option to create a more natural looking corner rounding. This option increases the corner radius at obtuse angles, and reduces the radius at acute angles, yielding a more natural shape.

From left to right: original shape, rounded shape, and rounded shape with visual corrections.



## 4.7 TRANSFORMATIONS

### 4.7.1 Transform

Use Transform to horizontally and vertically move, scale, and skew outlines. For scaling and skewing, you can set the transformation origin to baseline, x-height, cap height, half x-height and half cap-height. Skew without optical correction by activating the *Slant* option, or with optical correction using the *Cursify* option. Cursify requires correctly set vertical and horizontal stems in the *Masters* tab of *File > Font Info*.

### 4.7.2 Background

Use the slider in the Background tab to interpolate between front and background paths. The paths need to be compatible. This is useful if you want to experiment with a feature of a glyph, e.g., the length of a stroke. You can quickly copy a path into the background by choosing *Layers > Selection to Background* (Cmd-K).

### 4.7.3 Metrics

In the Metrics tab, you can set the width and sidebearings of all selected letters at once. With the *Relative* option, the values will be added or subtracted.

## 4.8 THIRD-PARTY FILTERS

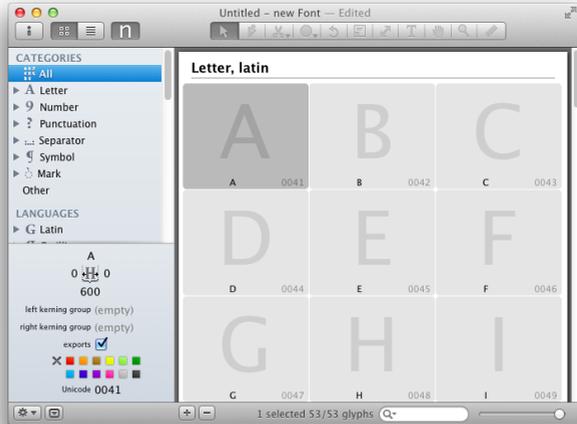
You can extend the functionality of Glyphs by installing additional filters. You can do so by double-clicking the file carrying the 'glyphsfilter' suffix. Glyphs will automatically put it in the correct folder, but needs to be restarted once afterwards. The filter will then show up in the *Filter* menu.

You can uninstall a filter by opening the Application Support folder for Glyphs and moving the respective files out of the folder called 'Plugins'. A quick way to navigate to that folder is *Script > Open Scripts Folder*. The 'Plugins' folder is located on the same level as the 'Scripts' folder.

# 5 Font View

## 5.1 VIEWING GLYPHS

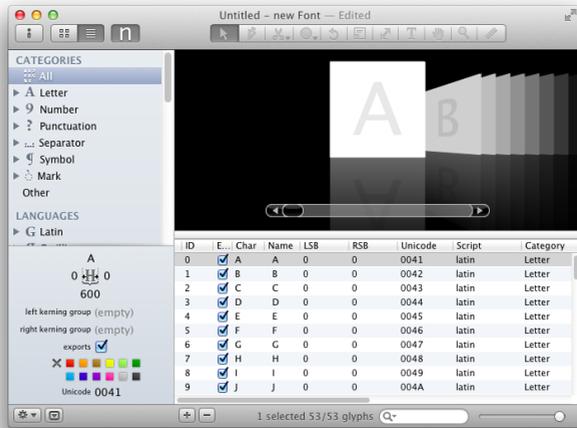
### 5.1.1 Grid View



By default, a new font contains uppercase A-Z, lowercase a-z and space. You can choose to view different excerpts of your glyphs by selecting categories, languages or filters from the left. Alternatively, you can enter a search term in the search field at the bottom right of the window (Cmd-F). By clicking on the magnifying lens, you can choose to search for unicodes instead of glyph names. With the slider next to it, you can control the zoom level of the displayed glyphs.

The three numbers to the left indicate (from left to right) how many glyphs are selected, how many are currently displayed, and the total number of glyphs in the font file.

## 5.1.2 List View



You can switch the Font view to a list mode by clicking the button with the list icon in the top left corner of the window:



In list view, each column represents a glyph property. You can sort glyphs by any of their displayed properties if you click on the corresponding column header. Click again to reverse the sort order. Click and drag a header to rearrange columns. Via the context menu of the column headers, you can control which columns are displayed.

## 5.2 ADDING GLYPHS

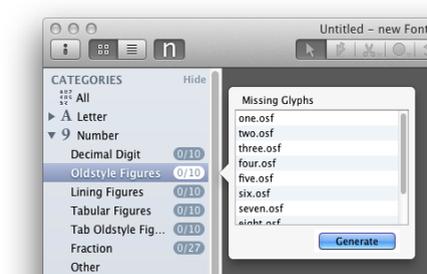
### 5.2.1 Generating New Glyphs

There are several ways to add new glyphs to a font:

- *Font* > *New Glyph* (Cmd-Opt-Shift-N) and the Plus button at the bottom in Font view add an empty glyph called 'newGlyph'. You can change its name by clicking on it.
- *Font* > *Duplicate* (Cmd-D) duplicates the currently selected glyph(s). The new glyphs will carry a name extension like '.001', '.002' etc.

- *Font > Add Glyphs* (Cmd-Shift-G) opens a dialog where you can insert a list of whitespace-separated glyph names. Glyphs has a built in list of composition recipes (see *Window > Glyph Info*) for component-based glyphs. You can control the composition of a glyph by entering custom recipes: ‘a+dieresis.alt=adieresis’ builds adieresis from ‘a’ and ‘dieresis.alt’ components, ‘tcommaaccent=tcedilla’ will create tcedilla with a single ‘tcommaaccent’ component. For a detailed description of component recipes, see section 2.4.4, ‘Components, Anchors, Clouds’ (p.17).
- Some entries in the sidebar of the Font view have a number badge, indicating the number of glyphs already created versus the total number of glyphs predefined in that category, language group or filter. Right-click or Ctrl-click it to get a list of all glyphs missing in this category. Select one or more glyph names, or press Cmd-A to select all of them, and click Generate to add them to your font:

For the complete list of recognized glyph names, choose Glyphs Info from the Window menu.



### 5.2.2 Copying Glyphs Between Files

You can copy any number of glyphs from one file (source) and paste them into another file (target). Components will be re-linked to their respective counterparts in the target font. If a glyph with the same name already exists in the target font, then the pasted glyphs will be renamed with an increasing triple-digit suffix, e.g., ‘A.001’, ‘A.002’ etc., avoiding data loss.

If you want to overwrite existing glyphs, you can use the *Paste Special* command from the *Edit* menu while holding down the Option key (Cmd-Opt-V). It appears if you hold down the Option key while opening the menu. A dialog will ask you for what precisely you want to do:

‘Overwrite glyphs with the same name’ just looks for

glyphs with the same glyph names as the ones pasted and will replace them.

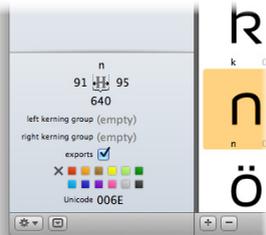
'Overwrite selected glyphs' will keep the target glyph names but replace their contents, including their layer structure. If the number of selected glyphs differs between copying and pasting, it will quietly ignore additionally selected target glyphs or additionally copied source glyphs.

'Only replace active layer' acts like the previous option, but it ignores all layers except for the one currently displayed.



## 5.3 GLYPH PROPERTIES

Select one or more glyphs and click on the second button  at the bottom left of the window, and the Glyph Properties will be displayed. Some of the properties can also be accessed through the context menu of one or more selected glyphs. In list view, glyph properties are displayed and can be edited in columns.



### 5.3.1 Name

The glyph name can be accessed and changed only if exactly one glyph is selected. If you want to batch-change the names

of many selected glyphs, you can use *Edit > Find > Find and Replace* (Cmd-Shift-F).

### **5.3.2 Width and Sidebearings**

The widths as well as the left and right sidebearings can be accessed and changed for any number of selected glyphs at once. Changing the width only affects the right sidebearing.

### **5.3.3 Kerning Groups**

You can set kerning groups for a range of glyphs, see chapter 7, ‘Spacing and Kerning’ (p. 63), for further details.

### **5.3.4 Exports**

The glyph will appear in the exported font only if this check box is enabled. This is useful for keeping glyph variations or glyph components from ending up in the final OTF.

### **5.3.5 Color Label**

For easier sorting and filtering, or simply for keeping oversight, you can mark glyphs with one of ten predefined colors. The colors have no effect on the exported font.

### **5.3.6 Unicode**

The Unicode is determined by the glyph name, which means that you cannot set the value directly. Thus, the displayed Unicode is read-only. However, if the *Don't Use Nice Names* option is activated in the *Other Settings* tab of the Font Info, you can set the Unicode manually.

### **5.3.7 Note**

In list view, you can display even more properties as columns. Right click on the column headers and tick off the columns you want to see. You can use them as sort key. Switch between ascending and descending sorting by clicking on the column title.

### **5.3.8 Components in List View**

In list view, you can have Glyphs display the elements of composite glyphs by right-clicking on the table header and activating Components in the context menu. You can change the composition of glyphs by changing their entries in that column, e.g., from ‘A, macron’ to ‘A, macron.case’.

### 5.3.9 Read-Only Properties in List View

The following properties can only be accessed in list view, they are read-only and only useful for sorting glyphs:

- **Unique Glyph ID** corresponds to the glyph order in the font; change this by setting the font-wide Custom Parameter 'glyphOrder' in *File > Font Info > Font*.
- **Character**, i.e., the Unicode character as represented by the OS X system font Lucida Grande or a fallback font.
- **Script**, e.g., 'Latin', 'Greek', 'Cyrillic'.
- **Category**, e.g., 'Letter', 'Number'.
- **Subcategory**, e.g., 'Uppercase'.
- **Last Changed**, i.e., date and time of the last manipulation of the respective glyph.

## 5.4 BATCH-PROCESSING

### 5.4.1 Layer Commands and Filters

You can apply manipulations on the active layers of a range of selected glyphs. Select the glyphs you want to batch-edit, and from the *Layers* menu, choose one of these commands:

- Make Component Glyph (Cmd-Opt-Shift-C)
- Decompose Components (Cmd-Shift-D)
- Add Image... (see 5.7, 'Images', p. 52)
- Update Metrics (Ctrl-Cmd-M)
- Assign Background ...
- Correct Path Direction (Cmd-Shift-R)
- Round Coordinates
- Tidy up Paths (Cmd-Opt-Shift-T)
- Add Extremes
- Set Anchors (Cmd-U)
- Reset Anchors (Cmd-Opt-U)

Apart from that, all Filters can be applied to more than one selected glyph.

### 5.4.2 Palette Manipulations

From the Palette, you can apply the mirroring, scaling, rotating, and skewing functions in the Transformations section on multiple selected glyphs.

## 5.5 FILTERING AND SORTING

### 5.5.1 Search Box

The search box (Cmd-F) at the bottom right provides a way to instantly narrow down the selection of glyphs displayed in the *Font* tab. Click on the zoom symbol in the search field to access additional options: you can choose to only search for a Unicode number or for a glyph name, or both; the *Match case* option activates the case sensitivity of the search term you enter.



### 5.5.2 Categories

Categories give you the option to only display a predefined subset of the glyphs you have. Click the triangle next to the Category name to access subcategories. You can combine more than one category by Cmd-clicking their names, e.g., if you Cmd-click the Categories 'Uppercase' and 'Lowercase', Glyphs will display both upper- and lowercase letters.

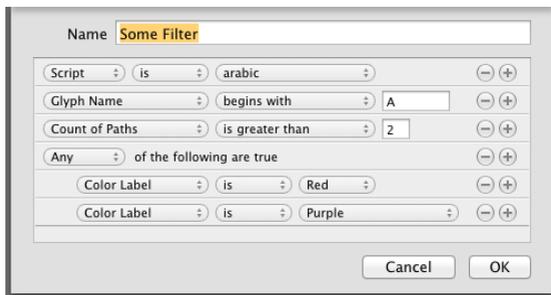
You can achieve an even narrower subset by also Cmd-clicking a Filter. For instance, if you click the 'Uppercase' Category and then Cmd-click 'MacRoman', Glyphs will display all uppercase characters in the MacRoman encoding.

You can cancel any subset you created and return to displaying the complete glyph set by clicking on the 'All' Category.

### 5.5.3 Languages

To only display glyphs belonging to a certain script, you can click on the name of the script in the Languages, e.g., 'Latin', 'Cyrillic', 'Greek' etc. Clicking on the triangle next to the script name will expand the view and give you more detailed subsetting options. Again, you can combine subsets by subsequently Cmd-clicking them.

## 5.5.4 Custom Filters



To define a query-based filter, click the gear button in the bottom left of the window and choose *Add Custom Filter*. Custom Filters are similar to smart playlists in iTunes. Set up the query with these properties:

- glyph name
- script (e.g., Latin or Arabic)
- count of paths (in the first master)
- compatibility of masters
- whether the glyph exports or not
- whether the first master has components or not
- color label

Add more properties by clicking on the plus button. By default, all properties are applied to the query (logical-and). You can add and nest logical-and ('all of the following') and logical-or ('any of the following') concatenations by holding down the Option key while pressing the button.

Once you have set up a couple of filters, you can apply more than one filter by holding down the Command key and subsequently clicking the individual Filter names.

## 5.5.5 List Filters

You can define a list of glyphs to be displayed by choosing *Add List Filter* from the gear menu in the bottom left of the window. By default, the selected glyphs are added to the list automatically. You can subsequently edit the list or paste a list of glyph names. In Font view, once you apply the List Filter, you will see its glyphs in the order they are listed in.

### 5.5.6 Custom Parameter ‘glyphOrder’

Categories and Filters have no influence on the order of the glyphs in the final font file. You can take control of the saving order with the ‘glyphOrder’ Custom Parameter in the *Font* tab of your Font Info. The parameter takes a comma-separated list of glyph names as value. If Glyphs finds this custom parameter, it will also display the glyphs in Font view in that order. When opening an OTF or TTF, Glyphs preserves the glyph order and creates the Custom Parameter automatically.

## 5.6 NAMES AND UNICODE

### 5.6.1 Readable or Nice Names

Glyphs contains a glyph info database that holds information about names, Unicode, components, anchors, and the like.

Not all glyphs that are defined in the Unicode database have official glyph names. For these, one would typically use names like ‘uni042F’ or ‘afii10049’, which are hard to memorize. Instead, Glyphs uses the Unicode description and appends a script suffix, e.g., ‘CYRILLIC CAPITAL LETTER A’ becomes ‘A-cy’.

The names, the Unicodes and some other info about the glyphs are defined in an XML file stored inside the application. You can extend or override the internal database with your own XML file in the Application Support folder for Glyphs. For details of managing your own glyph data, see: [glyphsapp.com/blog/roll-your-own-glyph-data/](http://glyphsapp.com/blog/roll-your-own-glyph-data/).

### 5.6.2 Naming Glyphs

The glyph name appears below the glyph in grid view, or in a separate column in list view. You can edit a glyph name by clicking once into the name. Glyphs will automatically convert entries like ‘ä’ or ‘uni00E4’ to its own naming convention (which is, for the Latin part, loosely based on the Adobe Glyph List Specification), i.e., ‘adieresis’ in this case. Also, Glyphs will automatically assign Unicodes and can even build OpenType features based on the glyph names. To see a list of built-in glyph names, choose *Window > Glyph Info*. Glyphs accepts three different types of input values:

- the ‘nice’ name, e.g., ‘Ia-cy’
- the Unicode value in hex form with a ‘uni’ or ‘u’ prefix, e.g., ‘uni042F’
- the Unicode character as typed from the keyboard, e.g., ‘Я’

For more information about the Adobe Glyph List, see [sourceforge.net/adobe/aglfn/home/Home/](http://sourceforge.net/adobe/aglfn/home/Home/)

This automation can be deactivated on a per-font basis: Go to *File > Font Info > Other Settings* and check the *Don't use nice names* option. If you do so, however, other automatic functions, like components and feature generation, will not work either.

For Unicode-based glyph names, use the prefix 'uni' followed by the four-digit hexadecimal code, e.g., 'uniE002'. For Unicodes outside the Basic Multilingual Pane (BMP), use 'u' as a prefix, followed by the five- or six-digit code, e.g., 'u10015'.

A valid glyph name must begin with a character from A-Z or a-z, but can subsequently also contain figures (0-9), underscores (\_), periods (.) or hyphens (-). All other characters, including whitespace characters, are not allowed. A dialog will appear if you try to use invalid characters like space in a glyph name.

If you need variations of glyphs, extend your glyph names with dot suffixes, like 'n.sc' for a small cap n, or 'five.sup5' for a superscript five. Some suffixes will be used by Glyphs to automatically build OpenType feature code. See the appendix of this manual for a list of recognized suffixes. Multiple suffixes should be added in the order of the OpenType features, that is, if you want to take advantage of the automatic feature generation. E.g., if you want to add a stylistic alternate for a small cap c, then you call it 'c.sc.sso1', since, by default, stylistic sets come after small caps.

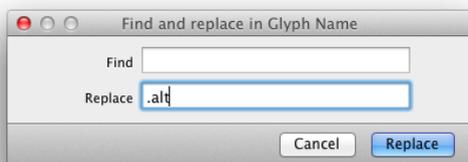
Hyphens are used to indicate the script a glyph belongs to, e.g., 'alef-ar' for an Arabic alef or 'ta-hiragana' for the ta syllable in the Japanese Hiragana script.

To name ligatures, the names of the parts need to be joined with an underscore, e.g., 'f\_f\_l' for an ffl ligature. The script suffix is appended only once (e.g., 'lam\_alef-ar'). Variation suffixes are only added once as well. These suffixes define the role of the whole ligature. For instance, a ligature formed of 'lam-ar.init' and 'alef-ar.medi' is called 'lam\_alef-ar.init'.

You can copy the names of selected glyphs into the clipboard by choosing *Copy Glyph Names* from the context menu. The names will be copied as a slash-delimited line of plain text, e.g., the names of a, ä, b, and c will be stored in the clipboard as: '/a/adieresis/b/c'.

### 5.6.3 Batch Renaming

You can search and replace in the names of selected glyphs via *Edit > Find > Find and Replace...* (Cmd-Shift-F). If you leave the Find field blank, then whatever is entered in the Replace field will be appended to the names of all selected glyphs. E.g., to quickly add an 'alt' suffix to all selected glyphs, you enter the suffix, including the dot, in the Replace field:



## 5.7 IMAGES

### 5.7.1 Adding Images

Like in Edit view, you can add an image to the current glyph layer by dragging it onto a glyph cell. By default, an image will be placed at the origin point of the currently active layer of the glyph in question, and scaled to where one DTP point corresponds to one font unit as defined by the Units per em setting in *File > Font Info > Font*. You can control positioning, scaling, and cropping of images in Edit view.

Batch-add images by choosing *Layers > Add Image...* and then selecting any number of images. Glyphs will place them in the appropriate cells based on their file name. E.g., an image called 'ntilde.png' will be put in ntilde, 'iacute.pdf' in iacute.

### 5.7.2 Viewing Images

Image display in Font view respects the *View > Show Image* setting unless there is no path drawn in the currently active layer of the glyph. In that case, the image will be shown regardless of the *Show Image* setting.

## 6 Font Info

### 6.1 FONT

Open the Font Info window by choosing *File > Font Info* (Cmd-I), or by clicking on the Info button in the top left corner of the main window. The *Font* tab contains information that applies to the whole font family. Info entered in the *Masters* tab or *Instances* tab takes precedence over the *Font* tab.

#### 6.1.1 Family Name

The name of the font family as it will appear in a font menu. Fonts carrying the same Family Name will be grouped in the same Style submenu. You can use a space in the font name, but non-ASCII characters may prevent the font from exporting.

#### 6.1.2 Units per Em

Number of units per em square (UPM), 1000 is the default. Increasing the UPM value can improve the representation of subtle details. The OpenType allows values up to 16,000 but values greater than 5000 can lead to problems in InDesign and Illustrator. Problems have been reported for other applications starting at 3000 UPM. Also, point coordinate values must not exceed  $\pm 32,768$ . If you need higher precision, it might be better to deactivate the grid by setting the Grid Spacing value to zero in *File > Font Info > Other Settings*.

Click on the double arrow next to the text field to scale the entire font. If you want to enlarge the font, set the UPM to something smaller than 1000, then scale back to 1000. Start with a higher value than 1000 to scale down. The quotient of these two values determines the scale factor.

**Tip:** to avoid rounding

errors, it may be a better idea to use the Scale to UPM parameter in an instance.

#### 6.1.3 Designer and Designer URL

Here, you can enter your name and a URL (including the protocol, e.g., <http://> or <ftp://>), for example:

- Designer: Jessica Doe
- Designer URL: <http://www.jessicadofonts.com/>

The entries correspond to OpenType Name IDs 9 and 12. You can check if the URL was entered correctly by clicking on the arrow next to the text entry field. Glyphs will open the URL in your web browser.

Name IDs refer to the entries of the OpenType Naming Table stored in OTF and TTF fonts. For a complete specification, see: [www.microsoft.com/typography/otspec/name.htm](http://www.microsoft.com/typography/otspec/name.htm)

#### 6.1.4 **Manufacturer and Manufacturer URL**

Here, you can enter name and URL (including the protocol, e.g., `http://` or `ftp://`) of your font vendor, for example:

- Manufacturer: Superfontstore
- Manufacturer URL: `http://www.superfontstore.com/`

The entries correspond to OpenType Name IDs 8 and 11. You can check if the URL was entered correctly by clicking on the arrow next to the text entry field. The URL will then be opened in your web browser.

#### 6.1.5 **Copyright**

A simple copyright notice. Click the circled arrow next to it to have Glyphs fill it in automatically. This will be recorded as Name ID 0.

#### 6.1.6 **Version**

Glyphs derives the Version string (Name ID 5) from the Version entry.

#### 6.1.7 **Date**

The creation date of the font. This entry will set the Creation and Modification dates in the OpenType Font Header table, also known as head table.

For more details about the head table, see: [www.microsoft.com/typography/otspec/head.htm](http://www.microsoft.com/typography/otspec/head.htm)

#### 6.1.8 **Custom Parameters**

Use entries in the Custom Parameter field to specify more settings. To add a parameter, click on the green plus symbol, select or type a property, and enter a value. You can also copy and paste parameters. You can use font-level custom parameters to override default values set by Glyphs, but are themselves overridden by master and instance parameters.

These properties are available font-wide:

- Add missing symbol glyphs
- blueScale
- blueShift
- description
- Family Alignment Zones
- fsType
- glyphOrder
- isFixedPitch
- license
- licenseURL

- panose
- ROS
- sampleText
- trademark
- unicodeRanges
- vendorID

See section 12.2, ‘Custom Parameters’ (p. 87) for possible values and a detailed description of custom parameters.

## 6.2 MASTERS

While information about the designed masters (the input) is set under ‘Masters’, the ‘Instances’ tab contains information about each instance that will be generated when fonts are exported (the output). Use the plus and minus buttons to add masters, and the ‘Add’ button to add any open single-master font as a master to the current font. You can add the font to itself to have two identical masters to start from.

Standard stems and alignment zones are necessary to optimize the font’s rendering on screen, where these characteristics may be represented by only a few pixels. Small deviations in the drawing can – after rounding to the pixel grid – result in considerable differences. Stems should have a uniform thickness. Alignment zones help create an even vertical alignment by overshoot suppression. The values entered here are crucial for the autohinting process when the font is being exported.

All values will be interpolated if entered in the same order throughout all masters. You can edit several masters at once after you Shift-click or Cmd-click the master names in the list on the left.

### 6.2.1 Proportions: Width and Weight

The values in these two pop-ups affect only the toolbar icons. The values for the actual interpolation are set under ‘Instances’. For more information about this, see chapter 9, ‘Multiple Master’ (p. 73).

### 6.2.2 Metrics

The vertical metrics have an impact on the line height, and serve as a guideline in the editing window. Glyphs will calculate the vertical metrics in the OS/2 and hhea tables from these values.

Also, Glyphs is able to find alignment zones (see below) automatically if you first enter correct vertical metrics: ascender, cap height, x-height, descender, perhaps also a `smallCapHeight` in the Custom Parameters.

When determining these values, ignore the overshoot. So, for instance, if you have the choice between various values for the x-height, say 490, 496 and 502, you want the one closest to your baseline, i.e., 490.

The Italic Angle has an effect on a number of tools throughout the application. Functions that respect the slant angle include aligning anchors between two selected nodes, the display of the x offset in measurement mode, and the calculation of the sidebearings.

### 6.2.3 Stems

If you enter good values for your standard stems, the autohinter will find those stems in your letters and put a hint on them. Try to find as few values as possible and as representative as possible, separate them by typing a comma. If you have a Multiple Master set-up, make sure the stems are listed in the same order in all masters, otherwise they cannot be interpolated correctly.

### 6.2.4 Alignment Zones

An alignment zone must encompass anything that should later be aligned at a low resolution. For instance, the upper edge of the crossbar of t, the upper edge of x and the highest extremum of o, they should be inside the alignment zone for the x-height. However, the size of an alignment zone must not exceed 25 (or -25 for bottom zones), and there must be at least one unit between any two alignment zones.

After you have set the Metrics properly, you can click the grey circle to let Glyphs find the alignment zones for you. Glyphs will then reduplicate the heights of the vertical metrics you entered in the Metrics field as the positions of the zones. It also respects the 'smallCapHeight' custom parameter if present. And it will try to guess the size of the alignment zones by measuring certain key glyphs in the font. If it cannot find any glyphs to measure, the sizes default to 16 for top zones, -16 for the descender zone and -15 for the baseline zone. It is a good idea to double check if your most important glyphs do end inside the alignment zones.

For a more detailed discussion of stems and zones, see the Hinting chapter.

### 6.2.5 Custom Parameters

Click on the green Plus button to add custom parameters for the masters. You can also copy and paste parameters. Values are interpolated if defined in all masters. Master parameters override all font-wide settings and parameters, but are overridden by instance-level settings and parameters. Glyphs accepts these master-level parameters:

- typoAscender
- typoDescender
- typoLineGap
- smallCapHeight
- winAscent
- winDescent
- vheaVertAscender
- vheaVertDescender
- vheaVertLineGap
- hheaAscender
- hheaDescender
- hheaLineGap
- underlineThickness
- underlinePosition

See section 12.2, 'Custom Parameters' (p. 87), for possible values and a detailed description of custom parameters.

## 6.3 INSTANCES

Use the plus and minus buttons in the lower left corner of the Font Info window to add and remove instances. You can edit several instances at once if you Shift-click or Cmd-click the instance names in the list on the left.

### 6.3.1 Style Name

This is the style name as it will appear in InDesign's font menu. You can use a space in the style name, but non-ASCII characters may prevent the font from exporting.

### 6.3.2 Weight and Width

These settings affect the order of fonts in the font menu. Sorting occurs first by width, then by weight, e.g.:

- Condensed Light
- Condensed Regular
- Condensed Bold
- Light

- Regular
- Bold
- Extended Light
- Extended Regular
- Extended Bold

### 6.3.3 Style Linking

Microsoft Windows applications support only the basic four styles per font family, i.e., Regular, Italic, Bold, and Bold Italic. They are accessed via the 'B' and 'I' buttons in the toolbar of the Windows application. Mac applications can switch to linked Bold and Italic styles via the Cmd-B and Cmd-I shortcuts. Adobe InDesign does the same via Cmd-Shift-B and Cmd-Shift-I. If a font family is not style-linked at all, then each style appears individually in the font menu. In that case, clicking the 'B' or 'I' buttons may create a synthetic Bold or Italic. Based on these conditions, the following strategy is recommended:

- The Bold, Italic, and Bold Italic styles should always be linked to the Regular.
- Other Italic styles should always be linked to their upright counterparts, e.g., Medium Italic is the Italic of Medium. They must share the same family name.

Some designers link the Semibold to the Light. If you choose to do so, only the Light will appear in the font menu of some applications and users may be unaware that they can access the Semibold style by selecting the Light in the menu and then activating the Bold style. If you still do want to link Semibold to Light, enter the name of your Light style in the Style Linking text field of your Semibold instance settings and check the Bold option next to it.

### 6.3.4 Interpolation

The Weight and Width settings apply to the design space spanned between the Masters. For further details, see chapter 9, 'Multiple Master' (p.73).

### 6.3.5 Custom Parameters

With instance-level parameters, you can produce different versions of the same font carrying different copyright notices, UPM values etc. Instance parameters override all master and

font settings and parameters. The following values are being considered during export:

- Add missing symbol glyphs
- compatibleFullName
- familyName
- fileName
- Filter
- hheaAscender
- hheaDescender
- hheaLineGap
- interpolationWeightY
- license
- licenseURL
- panose
- postscriptFontName
- postscriptFullName
- preferredFamilyName
- preferredSubfamilyName
- Remove Features
- Remove Glyphs
- Rename Glyphs
- sampleText
- Scale to UPM
- styleMapFamilyName
- trademark
- typoAscender
- typoDescender
- typoLineGap
- underlinePosition
- underlineThickness
- unitsPerEm
- vheaVertAscender
- vheaVertDescender
- vheaVertLineGap
- weightClass
- winAscent
- winDescent
- WWSFamilyName
- WWSSubfamilyName

**Pro User Tip:** In complex Multiple Master set-ups, the Bracket trick described in the section 3.4.2, 'Switching Glyph Shapes' (p.36), does not work. In that case, create copies of the glyphs (e.g. dollar and cent) and add a 'bold' suffix to their name (dollar.bold, cent.bold). In the bolder instances, add two custom parameters. First, use 'Remove Glyphs' for the original glyphs (e.g. 'dollar, cent'). Then add 'Rename Glyphs' to put the .bold glyphs in their place (e.g. 'dollar.bold=dollar, cent.bold=cent'). To be complete, you can use a custom parameter in your lighter weights to remove the .bold glyphs there.

See section 12.2, 'Custom Parameters' (p.87), for a detailed description of custom parameters.

## 6.4 FEATURES

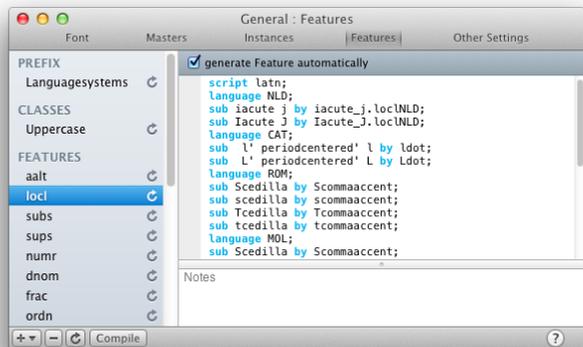
OpenType features entered in *File > Font Info > Features* apply to the whole font. Specific features can be disabled on a per-instance with the Remove Features parameter.

Glyphs automatically generates most of the common features. Click the circled arrow in the lower left corner to initiate the automatic generation of the feature code. The feature generator is based on certain name conventions and suffixes. E.g., all letters with names ending in 'sc' will be put in the small caps features. Find a complete list of recognized suffixes in section 12.1, 'Automatic Feature Generation' (p. 84).

To write features, prefixes, and classes manually, uncheck the 'Automatic' checkbox, or create a new item by clicking the plus button at the bottom left. Edit the name of the feature, prefix, or class directly in the list. Glyphs uses Adobe FDK syntax. You can test-run your features with the 'Compile' button. The prefix is for all information that needs to stand outside an actual feature, e.g., lookup definitions you want to re-use further down in the code. Glyphs will put language systems into the prefix.

The bottom right area is for notes. In Stylistic Set features (sso1 through sso20), you can use it to set the name of the Set. Type 'Name:', followed by the intended description. Alternatively, you can enter the complete featureNames lookup as described in the Adobe FDK syntax.

The Adobe FDK feature file syntax is described in detail at:  
[www.adobe.com/devnet/opentype/afdko/topic\\_feature\\_file\\_syntax.html](http://www.adobe.com/devnet/opentype/afdko/topic_feature_file_syntax.html).



## 6.5 OTHER SETTINGS

### 6.5.1 Grid Spacing

The grid width defines how coordinates get rounded. Standard is the value 1. When set to zero, no rounding will happen.

Contrary to popular belief, decimal coordinates can be exported into OTFs.

This is useful if you want to keep very fine details, e.g., after applying the Hatch Outlines filter or in detailed dingbat fonts, or if you plan to scale your glyphs and want to minimize rounding errors. Higher values are helpful when creating a pixel font.

All tools and all modifications will consequently snap to the grid. Regardless of the grid settings, you can round all point coordinates of selected nodes or glyphs to integer numbers by choosing *Round Coordinates* from the *Layers* menu.

### 6.5.2 Subdivision

The Subdivision value gives you the option to subdivide the Grid Spacing if you need finer steps but want the larger grid spacing as design orientation. The value indicates into how many compartments the main grid is subdivided, e.g., a Grid Spacing value of 100 and a Subdivision value of 5 will yield a subgrid with a 20 units step.

### 6.5.3 Don't Use Nice Names

This option prevents the automatic replacement of glyph names. This may be of importance where special workflow requirements apply. The option is set in all fonts imported from OTF files and UFOs that are opened with Glyphs for the first time, if the *Keep glyph names from imported files* option is set in *Glyphs > Preferences > User Settings*. Deactivating this option does not immediately activate the automatic replacement of names. To trigger the automatic naming, use *Font > Update Glyph Info*. Attention: this may also invalidate the feature code.

Only when the *Don't Use Nice Names* option is activated, can you set your own Unicode values. Otherwise, Glyphs will derive them from your glyph names.

### 6.5.4 Disable Automatic Alignment

This option disables the automatic alignment of components and the autosync of metrics for component-based diacritics. You can still lock components via the context menu.

### **6.5.5 Keep Alternates Next To Base Glyph**

In Font view, glyph variations with name suffixes will stay next to the main glyph if this option is enabled. For example, 'h.ss16', 'h.alt', 'h.loclENG' will be displayed right after 'h' instead of being moved to the end of the category.

# 7 Spacing and Kerning

## 7.1 SPACING

Spacing is the process of adjusting the sidebearings to get an even rhythm of the text. There are no fixed rules for adjusting the white space. However, unless you are working on a monospaced font, the same shapes should have the same sidebearing, e.g., the D has the same LSB as the H, but its RSB is more like the O.

In Edit view, you can trigger the display of spacing information, including the rectangle that indicate the width and the vertical measurements, by choosing *Show Metrics* from the *View* menu (Cmd-Shift-M).

'LSB' stands for the left, 'RSB' for the right sidebearing of a glyph.

### 7.1.1 Spacing Shortcuts

There are keyboard shortcuts to change the spacing of the current letter in Text mode (T). Hold down the Ctrl key and use the left and right Arrow keys to change the LSB. Cmd and arrow keys changes the RSB. Hold down both Ctrl and Cmd keys to change both LSB and RSB simultaneously, effectively moving the glyph inside its width. Add the Shift key to manipulate in increments of 10 units.

Easy to memorize: the Ctrl key is located on the left, the Cmd key on the right. The keys are associated with the left and right sidebearing, respectively.

The shortcuts for changing the LSB collide with default shortcuts for switching between OS X Spaces. You can change or deactivate the Spaces shortcuts in the System Preferences.

### 7.1.2 Linked Metrics

Linked metrics work in a very similar way as kerning classes: Put in the name of the glyph you want to link to in the sidebearing field and it will automatically adopt the respective side bearing. Note that this is not updated automatically. You need to select the glyphs that need updates and select *Layer > Update Metrics*. Or update by clicking in one of the metric fields in the info box and then clicking somewhere else.

You can even enter simple calculations into the sidebearing field. Calculations need to start with an equals sign (=). For instance, '=n+10' will take the same sidebearing of n and add 10 units, '=n-10' subtracts 10 units, '=g/2' yields half the sidebearing of g, and '=v\*2' doubles the sidebearing of v.

Use the pipe character (|, Shift-backslash on a U.S. keyboard, Opt-7 on a German keyboard) to reference the opposite

sidebearing. E.g., in the RSB of e, you can enter ‘=|a’ to use the LSB of a for the RSB of e.

## 7.2 KERNING

Most glyphs fit well next to each other but some pairs need specific adjustments, e.g., T in combination with o. This is where kerning comes into play.

### 7.2.1 Ways to Kern

To define these kerning pairs, switch to the Text Tool, type both glyphs in the Edit view and place the cursor between them. The bottom left field (labeled ‘K:’) in the info box will show the kerning value for this combination. Just set a value there.

Or use the keyboard shortcuts for much greater convenience. Similar to the metrics keys, Ctrl-Opt-(Shift-) Arrow keys changes the kerning towards the letter on the left side of the current letter, while Cmd-Opt-(Shift-)Arrow keys change the kerning with the letter following on the right.

**Tip:** Again, Ctrl (to the left of the Option key) corresponds to the left side of the glyph, Cmd on the right to the right side.

### 7.2.2 Kerning Groups

Many glyphs look similar and need the same kerning values. Kerning groups capture these similarities and help you reduce the number of pairs that need to be set manually. Kerning then applies not to glyphs, but to groups of glyphs, in the sense of kerning all glyphs that look like an A or like a T etc.

In Glyphs, kerning classes are not edited as glyph lists. Instead, the class membership is defined as a glyph property. For example, put O in the left kerning group field for all glyphs that look like an O on the left (like C, Ccedilla, G, Odieresis, and O itself). Add a value to each glyph, even if it remains the only glyph in its group. Switch to the list mode of the Font view for a better overview.

**Tip:** The ‘Set Kerning Groups’ script from [github.com/schriftgestalt](https://github.com/schriftgestalt) can create the kerning groups for you.



**Tip:** To ensure proper kerning interpolation in Multiple Master set-ups, the group locks need to be synchronized between all masters.

You can introduce exceptions to group kerning by opening the kerning lock in the grey info box. Thus you can have different kerning pairs for ‘To’ and ‘Tö’. Vice versa, keep these locks closed if you do not want any exceptions.

Kerning exceptions are handled per glyph. Move the cursor to the left of the glyph you want to define an exception for and open the lock. Close the lock again to remove the exception for that glyph. Within one pair, the kerning can be an exception for one glyph while it applies to the whole group for the other glyph. E.g. the kerning value defined for 'Tö' can be an exception for ö, since it does not apply to the other o glyphs. Thus, for ö, the left lock is open. However, the value does apply to all T's including variations like T̄ or Ṫ. So, for T, the right lock is closed.

### 7.2.3 Viewing Kerning Pairs

The Kerning window (*Window > Kerning*) lists all available kerning pairs, sorted by the first glyph or glyph class of the pairs. Clicking on a pair displays it in the current Edit tab. Make sure the Kerning button  in the bottom right corner of the window is on. Groups are marked with an at sign, e.g. '@A' for the A kerning group, and displayed in dark blue. Individual letters are displayed in beige.

You can color mark the kerning between glyphs in a sample text in Edit view if you display the measurement line. For more details, refer to section 2.7, 'Measuring' (p. 23).

### 7.2.4 Deleting Kerning Pairs

Clicking on the Minus button in the bottom left corner of the Kerning window deletes all currently selected kerning pairs.

### 7.2.5 Cleaning up Kerning

After removing glyphs or after importing, invalid or impossible kerning pairs may be left. You can remove them with the Clean Up function from the gear button in the Kerning window.

### 7.2.6 Compressing Kerning

As soon as you have kerned a few individual pairs and have set kerning groups, you can take the kerning onto the group level with the Compress function of the gear button in the Kerning window. The function turns individual kerning pairs into group kerning pairs, employing the groups the individual letters belong to. Compressing kerning will keep explicit kerning exceptions that differ from the group kerning, but will

remove exceptions that have the same value as corresponding group kernings.

### 7.2.7 Adding an Additional 'kern' Feature Lookup

In the Features section of the Font Info window, you can add a feature called 'kern'. Whatever you enter here will be added as a separate lookup called 'kernCustom' at the end of the kern feature in the features file. Since it is own font-wide feature code, this additional kerning is not interpolated, but simply added to existing, interpolated kerning pairs. This is a good place for contextual kerning, which sometimes is needed for punctuation or situations where negative kerning would otherwise eat up the wordspace, e.g.:

```
pos f' 60 space [T V W];  
pos L' -100 quoteright' -50 A;
```

In the first example, the f-space pair receives an extra 60 units only before T, V, W. In the second line, the combination LA is treated as follows: the quoteright is pushed into the L by 100 units, the A is also moved to the left, but only by 50 units. This only happens in this exact constellation. Thus, the second line does not affect the combinations L'O or l'A.

## 7.3 SAMPLE TEXTS

The predefined sample texts cover the most important combinations in Latin fonts. Select *Edit > Select Sample Text...* (Cmd-Opt-F) and choose a line. Use the Arrow keys to move the cursor and the Enter key to close the dialog. You can define your own texts in the application preferences.

### 7.3.1 Placeholders

If you want to space a lot of letters, you can use dynamic placeholders in the sample text. Placeholders always show the current glyph. This can come in handy when you step through your glyphs and want to see how each of them looks in different situations or as a double letter. In an Edit tab, switch to the Text tool and choose *Edit > Add Placeholder* (Cmd-Opt-Shift-P) to insert a placeholder.

**Tip:** quickly advance to the next glyph by typing the Home and End keys (Fn and left and right arrow keys on a MacBook keyboard).

## 8 Hinting

### 8.1 FONT-WIDE HINTING SETTINGS

For an in-depth discussion, see: [typophile.com/files/hinting.pdf](http://typophile.com/files/hinting.pdf), [partners.adobe.com/public/developer/en/font/T1\\_SPEC.PDF](http://partners.adobe.com/public/developer/en/font/T1_SPEC.PDF) (specifically pages 35 – 45) and [vimeo.com/38364880](http://vimeo.com/38364880).

Before you do glyph-level hinting, you need to define a set of parameters that apply to all hinting throughout the font. These font-level hints are stored in the so-called ‘PostScript Private Dictionary’ inside the exported font. This is why sometimes, these settings are referred to as ‘private parameters’.

#### 8.1.1 Standard Stems

The autohinter needs good standard stem values in order to recognize the stems and apply hints automatically. And the screen rasterizer can make use of these values to optimize the pixel rendering, especially synchronizing stem thicknesses across the whole font at low resolutions. Try to find representative values for your horizontal and vertical stem widths and enter them in the *Masters* tab of *Edit > Font Info* (Cmd-I). Theoretically, up to twelve width values can be considered for each direction, but try to use as few values as possible. Best practice is to use only two vertical values, one for lowercase and one for uppercase, e.g., the stem widths of lowercase *i* and uppercase *I*, or a representative value between the stem widths of *i/I* and the bowl widths of *o/O*. The same applies to horizontal stems, like the horizontal bars of *t/T*, *f/F*, *e/E* etc. The first value you enter is the most important one. Use a value that represents your most-used glyphs, typically the lowercase letters. Any values that follow are used for the *StemSnapH* and *StemSnapV* parameters, respectively.

You can quickly measure the thickness if you select two nodes and take a look at the grey info box (Cmd-Shift-I). For instance, if you measure 68, 71, 72, 74, 75, 82, 83, 85 for your vertical stems, then you would pick 72 and 83 as standard vertical stems, because these are good median values for most of the stem measures.

In a Multiple Master set-up, values in individual masters are interpolated, so make sure you enter the values in the same order in each master.

#### 8.1.2 Alignment Zones

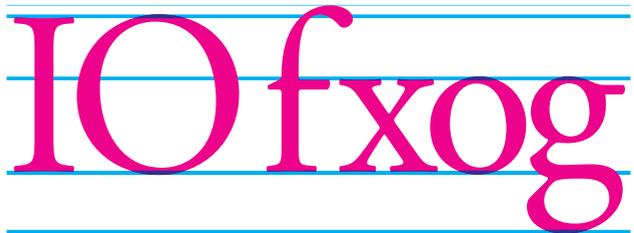
When your font is rendered with very few pixels on a computer screen, all the x-heights should align to the same

height, i.e., use the same amount of pixels vertically. The same applies to ascenders of letters like f, h, or k, descenders of g, p or y, and the heights of all capital letters. And of course, all letters should share the same baseline when pixellated at a low resolution.

But all these letters usually do not really align precisely. For instance, the bottom of a lowercase o will extend slightly below the baseline, while the serifs of an n may sit exactly on it. Or the apex of an uppercase A may extend a little bit beyond the height of an uppercase H. This difference, usually some ten units, is commonly referred to as 'overshoot'. Alignment zones are a way to tell the rasterizer about the overshoots. Anything with a valid horizontal hint that reaches into the same alignment zone will be aligned at low resolutions. In other words, the overshoots will be suppressed in small pixel sizes.

Alignment zones take two values: a position and a size. The position is the vertical height of the zone, usually the vertical metrics, like x-height or ascender. The position is sometimes also referred to as the 'flat edge' of a zone. The size is the thickness of the maximum overshoot that may appear at that position. If the overshoot extends above the position (e.g., x-height, cap height, ascender), the size value must be positive. Such zones are referred to as 'top zones'. If, however, the overshoots extend below the position (e.g., baseline, descender), the size must be negative and we call them 'bottom zones'.

A typical alignment zone setup: top zones with positive widths at ascender, cap height and x-height; bottom zones with negative widths at baseline and descender.



More precisely, the maximum size of an alignment zone is constrained by the blueScale value (see below), which implies that no zone must be larger than  $240 \div (240 \times \text{blueScale} - 0.98)$ .

Alignment zones should be as small as possible, so do not try to make them larger 'to be on the safe side'. In any event, a zone must not be larger than 25 units. You can have a maximum of 5 top zones and 6 bottom zones. Zones must not overlap. There must be a minimum distance of one unit between them.

### 8.1.3 Custom Parameters

Apart from the alignment zones and standard stems, there are more optional parameters in the Private Dictionary: `blueScale`, `blueShift`, and Family Alignment Zones. In Glyphs, you can set these values as Custom Parameters. For a detailed discussion of what they do, please see section 12.2, 'Custom Parameters' (p. 87).

## 8.2 AUTOHINTING

If the font-wide parameters are set properly, you can let the autohinting algorithm do its magic by simply activating the *Autohint unhinted glyphs* option in the export dialog. Test the hinting in an Adobe application. Write a test text and zoom out far enough so that the letters are displayed with a few pixels only. Then zoom in using the operating system's zoom function, which you can activate in the Accessibility settings of the System Preferences. If necessary, tweak your font settings or manually hint a problematic glyph and re-export. For details on manual hinting, see section 8.3, 'Manual hinting' (p. 70).

### 8.2.1 Flex hints

If the font has cupped serifs or slightly tapered stems, the autohinter can automatically apply so-called flex hints. Flex hints suppress the display of such shallow curves at low resolutions. You cannot manually set flex hints, they are automatically applied when the font is exported. In order for flex hinting to kick in, a few conditions must be met.

First the `blueShift` value must at least be set to the depth of your cups plus one. E.g., if your serifs are cupped 5 units deep, `blueShift` should be set to 6 or more.

Secondly, there are a few outline requirements. The cup or tapering must be built from exactly two consecutive outline segments between three nodes. The segments do not need to be symmetrical. The first and third node must share the same x coordinate (for tapered stems) or the same y coordinate (for cupped serifs). The four handles need not be completely horizontal (serifs) or vertical (stems), but the three nodes must be placed on the extremes of the two segments. The overall depth must not exceed 20 units.

Thirdly, in the case of cupped serifs, it is recommended that the three points are completely submerged in the respective

alignment zone. For best results, the second node (in the middle) should be exactly on the position (the 'flat edge') of the zone.

Flex hints: Nodes 1 and 3 are on the same level and inside the alignment zone, node 2 should be exactly on the flat edge of the zone. The handles must stay inside the space defined by nodes 1 through 3.



## 8.3 MANUAL HINTING

Manual and automatic hinting cannot complement each other inside a single glyph. Any manually hinted glyph is excluded from the autohinting process. Thus, if you decide to add hints manually, you must fully hint the glyph. Caution: Hint replacement is currently not supported, so you must make sure that there are no overlapping hints. A vertical hint may overlap a horizontal hint, but a vertical hint cannot overlap another vertical hint, and a horizontal hint must not interfere with another horizontal hint.

There are two types of glyph-level hints you can add manually, stem hints and ghost hints. Stem hints describe a vertical or a horizontal stem or stem-like feature of a glyph, like a serif or a crossbar. Ghost hints mark vertical edges when a horizontal stem hint cannot be applied.

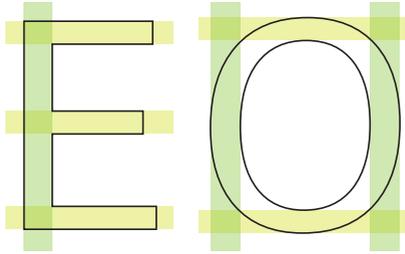
In combination with alignment zones, horizontal hints are important for the vertical alignment at the vertical font metrics, like the x-height or the ascender. At low resolutions, the rasterizer will try to vertically align the edges of all hinted horizontal stems that reach into an alignment zone. The horizontal hints must have their y coordinates in common with the nodes that are supposed to align. Since hints should not overlap one another, a single hint will do for all nodes it touches at its height.

### 8.3.1 Stem Hints

To add a stem hint to a glyph, right-click and choose *Add Horizontal Hint* or *Add Vertical Hint* from the context menu. A grey bar with a number badge will appear. The numbers indicate the origin (first number) and width (second number) of the hint. If you add a hint while two nodes are selected, the hint will be linked to these nodes right away. Adding linked

hints this way even works on multiple node pairs at once, as long as each pair is on a separate outline.

Positioning of vertical stem hints (green) and horizontal stem hints (yellow).



Select a hint by clicking on its grey number badge. Shift-click to select multiple hints. You can then edit the values numerically in the grey info area (*View > Show Info*, Cmd-Shift-I).

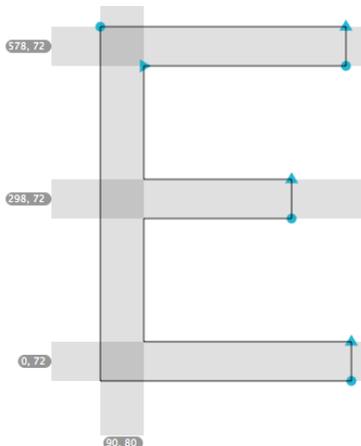
A horizontal stem hint, its number badge indicating a position of 140 and a width of 40.



To edit a hint graphically, drag the blue marks at the edges of the hint. The blue circle indicates the hint origin, while the triangle shows the size and orientation of the hint. If you drag one of the markers onto a node, Glyphs will link the hint to the position of the node. If you later move the node, the hint will adapt accordingly. You can delete one or more hints by selecting them and pressing the Backspace or Delete key.

**Pro User Tip:** In a Multiple Master set-up, if you connect all hints to nodes this way, you only need to hint the first master.

Hints linked to nodes with the blue triangle and circle. Stem hints must be positive, i.e., the triangle must be to the right or above of the circle.

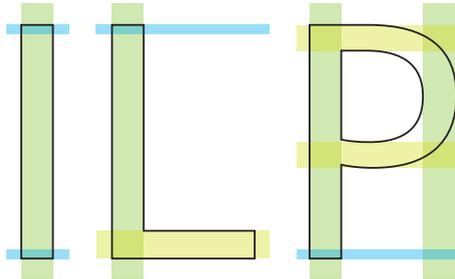


In the final OTF, all PostScript stem hints must be positive, i.e., have a width greater than zero. But even if you mistakenly insert a negative hint, Glyphs will correct its direction at export time, when all stem hints are turned positive.

### 8.3.2 Ghost Hints

You can use ghost hints when you need to vertically align the top or bottom of a glyph but cannot apply a horizontal hint. Take, for instance, a sans-serif uppercase I. The top needs to align with the cap height zone, the bottom with the baseline zone. In a serif I, you would apply horizontal hints to the serifs, but the sans-serif letter lacks the horizontal features necessary for a horizontal hint. In this case, you need to put a top ghost hint on the top of the I, and a bottom ghost hint at the bottom.

Positioning of ghost hints (blue) alongside regular stem hints.



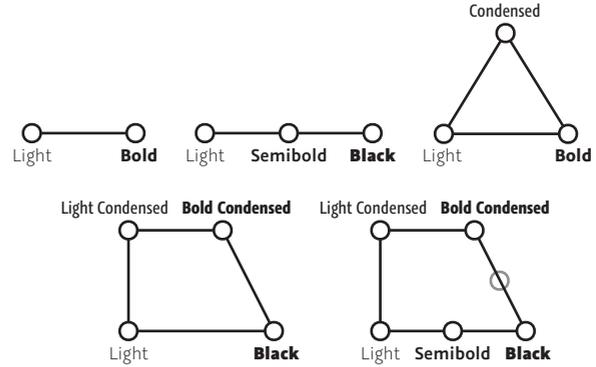
A top ghost hint at position 100, and a bottom ghost hint at position 50.



# 9 Multiple Master

## 9.1 OVERVIEW

**Tip:** It is advisable to keep uprights and italics in different files since their basic shapes are too different to linearly interpolate between them. See section 9.6, ‘Ensuring Family Consistency across Files’ (p. 76).

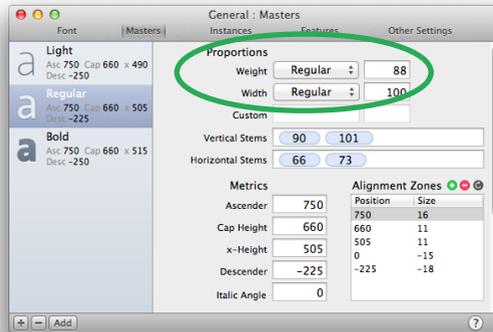


The relation between the masters is defined by their coordinates, i.e., their width and weight properties. In addition, you can define other dimensions. The masters span a design space in which the instances can be placed.

The nodes of the outlines are not automatically linked between the masters. You can insert nodes in one master while all other masers stay untouched. Note that for the final interpolation, the masters need to be compatible, though.

## 9.2 SET-UP

Defining the coordinates of the masters sets up the design space. For the Weight coordinate, it is recommended to use the stem width of a representative letter such as the n. This makes it easier to define the stem widths of the instances. The Weight coordinate is entered in the text entry field next to the pop-up.



Use the pop-up list to set the name of the master and the icon in the toolbar. This setting is not used for the style of the exported font; it is a rough guideline that allows for convenient buttons. The styles of the final fonts are defined in the instances.

## 9.3 MERGING TWO FILES

You can merge two or more open files into one Multiple Master File. To do so, activate the lighter font, go to *Font Info > Masters* and click *Add* in the bottom left corner. Select the other font in the upcoming dialog and click *OK*. Repeat this for all files you want to add, then set weight and width for all masters.

## 9.4 FIX OUTLINE INCOMPATIBILITY

When merging masters, Glyphs does not automatically fix incompatible outlines. This has to be done manually. Letters with incompatible outlines are marked with a pale red triangle in the Font view and a red stripe in Edit view. During editing, it is possible to work with incompatible paths but for interpolation, they need to be compatible. Specifically, this

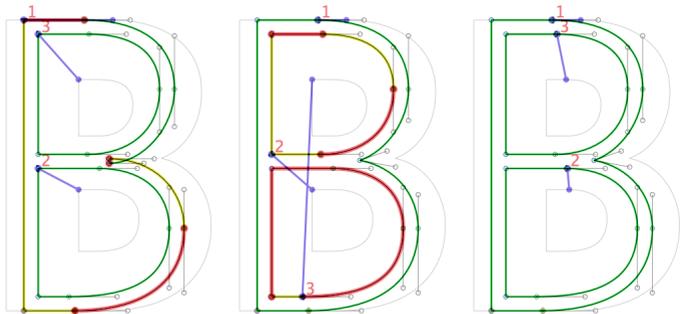


means that the order of the paths as well as number and structure of nodes must correspond with each other.

*View > Show Master Compatibility* (Ctrl-Command-Opt-N) activates a visualization of the masters' congruency. The red lines connect the corresponding nodes in all masters. If the picture looks as wild as in the first sample picture, then something is fundamentally wrong. The three slightly bigger red numbers indicate the paths' starting points and the path order. Straight blue lines connect the starting points of all masters. When comparing outlines, you always start at the start node of the paths and follow them in the direction that is indicated by the small triangle at the first node.

The colored segments indicate the compatibility of the segments. Red means that there is a line in one master and a curve in the other. These masters will not interpolate at all. Yellow means that the paths are of the same type but have a different angle. They will interpolate but may yield an unexpected result: differences in angles can turn smooth connections into so-called 'kinks', i.e., corners. Green segments are fully compatible.

Example:



**Tip:** Quickly switch between masters with Cmd-1, Cmd-2, etc.

In the foreground you see the colored light master and in background the bold master's paths, displayed in light grey. The foreground has an overlap, while the background has none. Choose *Filter > Remove Overlap* to get the second picture.

The outlines seem to fit now, i.e., they have the same number of nodes. The blue lines connecting the starting points in all masters are running across the letter though. The starting nodes have to be at the same position. Either

correct that manually by right clicking the node and choosing *Make Node First* from the context menu, or use the command *Layer > Correct Path Direction* (Cmd-Shift-R) on all masters.

In the third image, the start points are at the same position in both masters and all outlines are green.

## 9.5 LAYERS PANEL

The Layers Panel in the Palette (*Window > Palette*, Cmd-Opt-Shift-P) shows all layers of the selected glyph. The master layers are shown on top. You can copy them to save a working state: select the layer in the list and click the Copy button. When clicking an entry in the list, the selected layer is activated in the current glyph only. This is helpful to simultaneously see different layers of one glyph.



To change view between master layers, use the tool bar buttons. Also there is the shortcut Cmd-1, Cmd-2, etc., i.e., Cmd and the number of the respective master.

## 9.6 ENSURING FAMILY CONSISTENCY ACROSS FILES

If you extend your font family beyond the scope of a single Glyphs file, e.g., with italic styles, then make sure both files carry precisely the same Font Family Name in the *Font* tab of *File > Font Info*. Keep in mind that Font Family Names can be altered for an instance with a custom parameter. Except for a Bold Italic, italic styles always need to be style-linked with their non-italic counterpart in the *Instances* tab. The Bold Italic needs to be style-linked to the Regular. See section 6.3.3, ‘Style Linking’ (p.58), for more details.

To compare the glyph scope and glyph metrics of two or more open Glyphs files, choose *Edit > Compare Fonts...* Glyphs will then display a spreadsheet containing glyph counts, an indication which glyphs are missing in which file, and a per-glyph comparison of LSB, RSB, and width values for each Master layer. If you want to add the missing glyphs to one of the fonts, you can copy the names of the missing glyphs from the spreadsheet, then trigger *Font > Add Glyphs...* and paste the names into the appearing dialog sheet.

**Tip:** As of version 1.3.19, you do not need to remove the commas anymore from the Add Glyphs string.

# 10 Error Handling

## 10.1 GLYPH NAMES

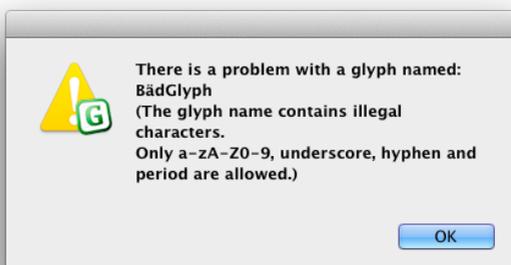
The most common source of problems are bad glyph names.

Make sure all your glyph names:

- only contain letters A-Z, a-z, numerals 0-9, underscore (\_), hyphen (-) or period (.),
- start with a letter (A-Z, a-z),
- have no whitespace characters (space, tab, return etc.), not even at the end,
- and contain no non-ASCII characters (like á, è, ß or ü).

Not adhering to these guidelines may yield an error message at export time like 'There is a problem with a glyph named:', followed by the bad glyph name and a descriptive explanation of the problem within brackets, e.g., 'The glyph name should not contain any space character.' Such error messages can also occur when trying to compile the OpenType feature code in the *Features* tab of *File > Font Info*.

You can find bad names by searching for space (or other invalid characters) in the search field of the Font view (Cmd-F).



## 10.2 FONT NAMES

If your error message only entails a POSIX path to a file called 'FontMenuNameDB', set between brackets (e.g., '[/Users/<your username>/Library/Application Support/Glyphs/Temp/<your fontname>/FontMenuNameDB]'), then there probably is an invalid character in the font name or style name. You can use spaces, but as of yet, no non-ASCII characters.

### 10.3 DUPLICATE UNICODE VALUES

In some fonts, two or more letters erroneously carry the same Unicode. If you try to import a font with double encodings, Glyphs will warn you. Selecting all glyphs and choosing *Update Glyph Info* from the *Font* menu will usually fix that problem.

**Tip:** If you do want to keep a glyph accessible via two different codes, it is better to create a duplicate glyph.

In the reverse situation, where one glyph sports two Unicodes, Glyphs will silently reset the Unicode value based on the glyph name. The usual suspects for this are Delta and Omega. Delta should only be U+0394, but sometimes also has U+2206, the code for the mathematical increment operator (a.k.a. Laplace operator, glyph name 'increment'). Omega, which should only be encoded with U+03A9, sometimes also sports U+2126, the codepoint for the Ohm symbol (glyph name 'Ohm').

### 10.4 OPENTYPE FEATURE CODE

If all your features are auto-generated, you can usually fix feature code troubles by re-compiling, i.e., clicking the Compile button in the Feature tab of the Font Info window.

Glyphs tries to pass on any FDK compilation errors in an error dialog. And it also tries to point you to where the problem occurred by reporting the name of the problematic feature and the line number. If it succeeds in doing so, the error dialog will sport a 'Show' button. Click on it and Glyphs takes you directly to the code problem. The following errors may occur:

*Contextual substitution clause must have a replacement rule or direct lookup reference.* You probably forgot or mistyped the word 'by' in a substitution feature.

*DFLT script tag may be used only with the dlft language tag.* You tried using a language tag without a preceding script tag, e.g., if you use 'language DEU;', there needs to be a 'script latn;' somewhere before in the feature.

*"Feature" statement allowed only in 'aalt' feature.* You most likely added your own feature code in the Prefix and forgot to close a feature properly before starting the next one.

*Glyph x not in font.* You tried referencing a glyph that does not exist. The error message will tell you which glyph it was looking for in vain. You probably mistyped the name of the glyph, accidentally deleted the actual glyph, renamed the glyph, or forgot to create the glyph in the first place. Sometimes, the automatically generated features are simply

Unnecessary kerning between two scripts (e.g. a Cyrillic and a Latin letter with each other) is ignored at export time, minimizing the risk for this error..

out of date. In this case, recalculating the features by clicking on the circled arrow button in the bottom left corner of the *Features* tab in *File > Font Info* will do.

*GPOS feature 'kern' causes overflow of offset to a subtable.* The kerning structure is too complicated and causes the glyph positioning table in the font to become too large. An OpenType table must not be larger than 64 kilobytes. Cleaning up and compressing kerning may help. Try the respective functions from the gear menu of the Kerning window (*Window > Kerning*). Also check for unnecessary kerning pairs. Very small values (below 5) are usually superfluous and can be deleted.

*Invalid token.* Most likely one of the class names starts with a whitespace character, or there is a whitespace between the at sign (@) and the class name in the feature code, or there is a whitespace character in a glyph name, or there are invalid non-ASCII characters elsewhere in the feature code.

*Lookup type different from previous rules in this lookup block.* You tried to mix contextual and non-contextual positioning rules, probably in the 'kern' feature.

*makeotfGlyphs.* The most probable cause is a glyph name containing invalid characters.

*makeotfGlyphs [FATAL] line too long.* Most likely, a glyph name was too long. Glyph names should not be more than 122 characters long.

*not in range -32767..32767 (text was "...").* At least one node or anchor in the font is out of bounds. A coordinate value must not exceed  $\pm 32,767$ . Open the Glyphs file in a text editor and search for the string 'e+'. This way you can find large numbers in exponential notation, e.g., '-9.22337e+18' and see which glyph needs to be fixed. If you do not find anything this way, try the number in the brackets of the error message.

*Positioning values are allowed only in the marked glyph sequence, or after the final glyph node when only one glyph node is marked.* The syntax of a contextual positioning rule in the 'kern' feature is faulty. At least one glyph name must be marked with a single dumb quote ('), and the number value should come right after the marked glyph, not at the end as in other positioning rules.

*Premature end of input.* Most likely, one of the glyphs has a bad name. Check if there is a glyph name that contains an equals sign (=), an at sign (@) or brackets ('[' or ']').

*Target glyph class in rule doesn't have the same number of elements as the replacement class.* You tried substituting a class with a class of a different size. The error dialog will point you to the problematic code line. Make sure the classes are of the same size and in the same order.

*Syntax error.* This can have several reasons:

- You are using non-ASCII characters in your code.
- One of the feature names does not adhere to the naming rules: exactly 4 lowercase ASCII characters and figures, no whitespace or punctuation.
- You mistyped feature commands like 'sub', 'pos' or 'by'.
- You forgot a semicolon at the end of a code line. In this case, the error message will contain 'missing ";"'.
- You forgot a numerical value in a positioning lookup. In this case, the error message will also contain 'missing NUM'.

## 10.5 MISSING OUTLINES

Sometimes the font does export, but some glyphs are empty or parts of glyphs are missing.

### 10.5.1 Open Paths

If the *Remove Overlap* option is active at export, outlines which are not closed are ignored and will not show up in the final font. So, if a glyph appears empty in the OTF, it is a good idea to check if its paths are actually closed.

### 10.5.2 Wrong Path Orientation

Outlines must be oriented counter-clockwise, except for counters, which need to be oriented clockwise. Otherwise, counters may appear 'closed' or missing. You can fix the orientation of selected outlines or even multiple selected glyphs with *Layers > Correct Path Direction* (Cmd-Shift-R).

### 10.5.3 Multiple Paths on Top of Each Other

Two similar paths on top of each other with varying path orientations may delete each other at export. You can select exactly one whole path by double clicking it. Press the Backspace key to delete it. If it looks like nothing has changed, then you had at least two identical outlines on top of each other.

#### **10.5.4 Outline Incompatibility**

Incompatible outlines should not hinder OTF export. Affected glyphs are simply exported empty. If you get a ‘Some glyphs are not compatible and will have no outlines’ error message, the font will still be exported. You can find more details about fixing incompatible outlines in section 9.4, ‘Fix Outline Incompatibility’ (p. 74).

# 11 Import and Export

## 11.1 FONTLAB

### 11.1.1 From FontLab to Glyphs

There is a Python script available to export Glyphs files directly from within FontLab. Point your web browser to [github.com/schriftgestalt/Glyphs-Scripts/](https://github.com/schriftgestalt/Glyphs-Scripts/) and download the script called 'Glyphs Export.py'. To install, invoke *Go > Go to Folder...* in Finder, enter `~/Library/Application Support/FontLab/Studio 5/Macros/` and place the file in the Macros folder that is displayed then. After restarting FontLab, it will become available in the macro toolbar.

### 11.1.2 From Glyphs to FontLab

Again, there is a Python script to import Glyphs files into FontLab. You can get it from the same Github repository mentioned above. This time, look for a script named 'Glyphs Import.py' and place it in your FontLab Macros folder.

## 11.2 ROBOFONT AND OTHER UFO TOOLS

Glyphs can read and write UFO files. Unfortunately, UFO files cannot contain more than one master. So, *File > Export* will export a Multiple Master Glyphs font as several UFO files.

When importing a font project, Glyphs will try to apply its built-in naming scheme and sync the metrics of compound glyphs with their base glyphs. To prevent either of the two, you can go to *Glyphs > Preferences > User Settings* and check the options 'Keep glyph names from imported files' and 'Disable automatic alignment in imported files'. To set these options on a per-font basis, go to *File > Font Info > Other Settings*, and check 'Don't use nice names' or 'Disable automatic alignment', respectively.

## 11.3 OPENTYPE AND TRUETYPE

### 11.3.1 Opening Existing Fonts

For more details on opening existing fonts, see [glyphsapp.com/blog/importing-existing-fonts](https://glyphsapp.com/blog/importing-existing-fonts)

While you can open existing OTF, TTC, and TTF fonts, Glyphs cannot reverse-engineer all the information inside a compiled font file. That means that opening an OTF and exporting it again will produce a file that is different from the original. For instance, you will lose all OpenType feature code, hints,

and some of the font metadata stored in OpenType tables. It is strongly advised to always work on a copy.

### 11.3.2 Generating Installable Fonts

Choosing *File > Export ...* will bring up the Export dialog. To generate an OpenType font for use in layout or word processing applications, pick OTF as export format.

The *Use production glyph names* option will automatically rename glyphs of the final font file according to the Adobe Glyph List. Some applications and systems, e.g., the text engine of OS X 10.4, expect this naming scheme. Not using this option may render the font incompatible with such environments, so uncheck it only if you know what you are doing.

The *Remove overlap* option applies the *Remove Overlap* filter at export. See section 4.4, '*Remove Overlap*' (p. 40), for more details. In a release font, overlaps must be removed. So, uncheck this only for testing purposes, or if you have already removed overlaps manually or with a custom parameter.

The *Autohint unhinted glyphs* option applies the AFDKO autohinting algorithm to all glyphs that do not have any manually set hints. This option expects standard stems and alignment zones to be set correctly (see 8, '*Hinting*', p. 67).

Currently, the *Save as TTF* option is only an experimental implementation, and does not yet produce TrueType fonts ready for release. TTFs generated this way need to be processed by a third-party application like FontLab.

The *Export destination* option allows you to set a default location the fonts will get exported to. If you do not set a destination, Glyphs will bring up a save dialog. Be aware that exporting your font will overwrite any previous instances with the same name in the same export location. This can be useful if you use the Adobe Fonts folder as export destination (see 2.10.7, '*Previewing in Adobe applications*', p. 32).

# 12 Appendix

## 12.1 AUTOMATIC FEATURE GENERATION

Glyphs can automatically generate a number of OpenType features if it finds glyphs with certain names in the font.

<b>aalt</b>	<i>All Alternatives</i>	Glyphs automatically builds the aalt feature based on all features that substitute glyphs.
<b>liga</b>	<i>Ligature</i>	Join the glyph names of the components with an underscore ('_'). Some common ligatures (f_f_i, f_f_l, f_f, fi, fl, lu_lakkhangyao-thai, ru_lakkhangyao-thai) are automatically placed in the liga feature, all others go into dlig by default. However, if you want to force a ligature into the liga feature, you can add a 'liga' suffix to its name, e.g., 'f_b.liga' or 'yi_yi-cy.liga'.
<b>dlig</b>	<i>Discretionary Ligatures</i>	Join the glyph names of the components with an underscore ('_'), e.g., 'f_odieresis'.
<b>rlig</b>	<i>Required Ligatures</i>	Add 'rlig' to the ligature name. Also triggered by lam_alef-ar, lam_alefHamzaabove-ar, lam_alefHamzabelow-ar, lam_alefMadda-ar, lam_alef-ar.fina, lam_alefHamzaabove-ar.fina, lam_alefHamzabelow-ar.fina, lam_alefMadda-ar.fina, lam_alefWasla-ar, lam_alefWasla-ar.fina
<b>czsc</b>	<i>Small Capitals from Capitals</i>	Add 'sc', 'czsc' or 'smcp' to the glyph name.
<b>smcp</b>	<i>Small Capitals</i>	Add 'sc', 'czsc' or 'smcp' to the glyph name.
<b>sup</b>	<i>Superscript</i>	Add 'sup' to the glyph name. Or extend figure names with 'superior', without the period, e.g., 'onesuperior'.
<b>sub</b>	<i>Subscript</i>	Add 'inferior' or 'sub' to the glyph name.
<b>sinf</b>	<i>Scientific Inferiors</i>	Add 'sinf' or 'sub' to the glyph name.
<b>dnom</b>	<i>Denominators</i>	Add 'dnom' to the glyph name.
<b>numr</b>	<i>Numerators</i>	Add 'numr' to the glyph name.
<b>frac</b>	<i>Fractions</i>	The frac feature is generated from the .numr, .dnom and fraction glyphs. If they are not present in the font, then the feature will be composed from any pre-built fractions available in the font, like onehalf, onequarter, threequarters etc.

If you have separate sets for czsc and smcp, you can use 'czsc' for uppercase glyph names and 'smcp' for lowercase glyph names.

If your font does not differentiate between subscript and scientific inferior, simply use one set of 'sub' glyphs and Glyphs will create both features with it.

**Tip:** do not use the figure suffix which would apply to your default figures: .lf, .tf, .osf, or .tosf. This way, Glyphs can determine what type your default figures (i.e., figures without suffix) are and build the feature code accordingly.

<b>onum</b>	<i>Oldstyle Figures</i>	Add 'osf' (for proportional oldstyle figures) or 'tosf' (for tabular oldstyle figures) to the glyph name. Can be applied to other characters as well, e.g., currency signs.
<b>tnum</b>	<i>Tabular Figures</i>	Add 'tf' (for tabular figures) or 'tosf' (for tabular oldstyle figures) to the glyph name. Can be applied to other characters as well, e.g., currency signs.
<b>pnum</b>	<i>Proportional Figures</i>	Add 'osf' (for proportional oldstyle figures) or 'lf' (for proportional lining figures) to the glyph name. Can be applied to other characters as well, e.g., currency signs.
<b>lnum</b>	<i>Lining Figures</i>	Add 'lf' (for proportional lining figures) or 'tf' (for tabular figures) to the glyph name. Can be applied to other characters as well, e.g., currency signs.
<b>ordn</b>	<i>Ordinals</i>	Automatically built if 'ordfeminine' and 'ordmasculine' are found in the font.
<b>ornm</b>	<i>Ornaments</i>	Add 'ornm' to the glyph name of letters A–Z or a–z. Also make sure you have the glyph 'bullet' in your font.
<b>hist</b>	<i>Historical Forms</i>	Add 'hist' to the glyph name.
<b>case</b>	<i>Uppercase Forms</i>	Add 'case' to the glyph name or 'lf' to the name of a figure.
<b>cpSP</b>	<i>Capital Spacing</i>	Add uppercase letters to your font.
<b>locl</b>	<i>Localized Forms</i>	Add 'loclXXX' to the glyph name, where XXX represents the three letter language tag, e.g., 'loclENG' for English. There also is a built-in list of glyphs which are added automatically to the feature: <ul style="list-style-type: none"> <li>• idotaccent, i.TRK or i.loclTRK: trigger i substitutions for TRK, AZE, CRT, KAZ and TAT if i is present.</li> <li>• Scommaaccent, Tcommaaccent, scommaaccent, and tcommaaccent: trigger substitutions for ROM and MOL if Scedilla, Tcedilla, scedilla, and tcedilla are present.</li> <li>• Ldot, ldot: trigger l geminada substitutions for CAT if L, l, and periodcentered are present.</li> <li>• Iacute_J.loclNLD and iacute_j.loclNLD, or Jacute and jacute: trigger accented ij substitutions for NLD if Iacute, iacute, J, and j are present.</li> </ul>

For a complete list of language tags, see: [microsoft.com/typography/otspec/languagetags.htm](https://microsoft.com/typography/otspec/languagetags.htm)

<b>ssXX</b>	<i>Stylistic Set</i>	Add ‘ss01’ through ‘ss20’ to the glyph name. If you add ‘Name:’ plus the feature’s descriptive name to the feature note at the bottom, Glyphs will generate the feature name entries for stylistic set names. Alternatively, you can add the complete featureNames definition to the note.
<b>salt</b>	<i>Stylistic Alternates</i>	By default, Glyphs will duplicate the ss01 feature in salt. Adobe Illustrator and Adobe Photoshop make use of this feature in their OpenType palettes.
<b>swsh</b>	<i>Swashes</i>	Add ‘swsh’ to the glyph name.
<b>titl</b>	<i>Titling</i>	Add ‘titl’ to the glyph name.
<b>init</b>	<i>Initial Forms</i>	Add ‘.init’ to the glyph name.
<b>medi</b>	<i>Medial Forms</i>	Add ‘.medi’ to the glyph name.
<b>med2</b>	<i>Medial Forms</i>	Add ‘.med2’ to the glyph name. Used only with the Syriac script.
<b>fina</b>	<i>Terminal Forms</i>	Add ‘.fina’ to the glyph name.
<b>fin2</b>	<i>Terminal Forms</i>	Add ‘.fin2’ to the glyph name. Used only with the Syriac script.
<b>fin3</b>	<i>Terminal Forms</i>	Add ‘.fin3’ to the glyph name. Used only with the Syriac script
<b>vert</b>	<i>Vertical Alternates</i>	Add ‘.vertical’, ‘.vert’ or ‘Vertical’ to the glyph name.
<b>akhn</b>	<i>Akhands</i>	Triggered by k-deva, j-deva, ssa-deva, nya-deva and k_ssa-deva, j_nya-deva.
<b>blwf</b>	<i>Below Base Forms</i>	Triggered by ra-deva, halant-deva and rashtrassign-deva.
<b>cjct</b>	<i>Conjunct Forms</i>	Triggered by Devanagari conjunct clusters.
<b>half</b>	<i>Half Forms</i>	Triggered by Devanagari half-form glyphs ending in ‘Halfform’ in conjunction with halant-deva.
<b>nukt</b>	<i>Nukta Forms</i>	Triggered by Devanagari nukta ligatures ending in ‘Nukta-deva’, in conjunction with the same glyphs without nukta.
<b>rkrf</b>	<i>Rakar Forms</i>	Triggered by Devanagari rakar ligatures in conjunction with the isolated glyphs and halant-deva.
<b>rphf</b>	<i>Reph Forms</i>	Triggered by ra-deva, halant-deva and reph-deva.

All quotes in this chapter are taken from the Microsoft OpenType specification at [microsoft.com/typography/otspec/](http://microsoft.com/typography/otspec/) unless stated otherwise.

## 12.2 CUSTOM PARAMETERS

Custom parameters have a property and a value. In this appendix, the properties are printed in bold. The short description next to it explains the respective values and the function of the parameter.

### 12.2.1 Glyphs-Specific Parameters

The following custom parameters, presented in alphabetical order, are specific to Glyphs.

**Add missing symbol glyphs** *boolean* Triggers the ‘makeotf -adds’ option. According to Adobe, ‘MakeOTF includes two Multiple Master fonts built-in, one serif and one sans-serif. With these it can synthesize glyphs that match (more or less) the width and weight of the source font. It requires the glyphs zero and O to be present in the font, in order to determine the required weight and width.’

Source: Adobe  
MakeOTF Manual

The parameter will add these glyphs if they are not present in your font: approxequal, asciiircum, ascitilde, at, backslash, bar, brokenbar, currency, dagger, daggerdbl, degree, Delta, divide, equal, estimated, Euro, fraction, greater, greaterequal, infinity, integral, less, lessequal, litre, logicalnot, lozenge, minus, multiply, notequal, numbersign, Omega, onehalf, onequarter, paragraph, partialdiff, perthousand, pi, plus, plusminus, product, quotedbl, quotesingle, radical, section, summation, threequarters.

**Family Alignment Zones** *list* This parameter can help create a more consistent screen appearance at low resolutions, even if the overshoots differ in the individual weights. It is a good idea to reduplicate the most important (or ‘dominant’) alignment zones of the most important font in your family, usually baseline, x-height and cap height of the Regular or Book. A rasterizer will then try to align all weights if the height difference between the individual weight and the family alignment is less than one pixel.

Above: without family alignment.  
Below: with family alignment.

*family alignment.*  
*family alignment.*

**fileName** *string* Name for the OTF file, without the ‘.otf’ ending.  
Gives you the chance to export two versions of the same font style name without the second file overwriting the first one.

**Filter** *string* Applies Glyphs filters to all glyphs at export. Currently, the Offset Curve, Remove Overlap, Roughen, and Round Corners filters are supported. The values are as follows:

- GlyphsFilterOffsetCurve;<x>;<y>;<make stroke>;<position>
- GlyphsFilterRoundCorner;<radius>;<visual correction>
- GlyphsFilterRoughenizer;<length>;<x>;<y>;<angle>
- GlyphsFilterRemoveOverlap;

The boolean values for <make stroke> and <visual correction> are 1 for yes and 0 for no.

**glyphOrder** *string* Sets the order of glyphs in both the working file and the final font. Glyph names need to be separated by newlines. You can copy and paste the content of a List Filter.

**InterpolationWeightY** *integer* Vertical interpolation value. In an instance, you can differentiate between interpolation along the x axis and interpolation along the y axis by introducing this custom parameter. For it to take effect, it must differ from the interpolation weight of the instance.

**Remove Features** *string* Will remove the features mentioned in the value. Takes a comma-separated list of feature codes as value. Useful for disabling some OpenType features in a specific instance.

**Remove Glyphs** *string* Will remove the glyphs mentioned in the value. Takes a comma-separated list of strings as value. Useful for preparing the swapping of glyph shapes with the Rename Glyphs parameter (see below).

**Rename Glyphs** *string* Will rename the glyphs mentioned in the value. Needs a comma-separated list of rename strings of the form ‘oldname=newname’, e.g., ‘e.bold=e, eacute.bold=eacute’.

**ROS** *string* A special-purpose Adobe-Identity-0 ROS (Registry, Ordering, Supplement) for fonts with a Character To Glyph Index Mapping Table (cmap). If you do not plan to use one of the public ROSes (Adobe-CNS1-6, Adobe-GB1-5, Adobe-Japan1-6, Adobe-Korea1-2), this parameter allows you to insert a ROS you have built yourself.

**Scale to UPM** *integer* Scales the whole font to the supplied integer value. Useful for preparing a TTF conversion by scaling to 2048, for instance.

**smallCapHeight** *integer* A vertical metric for small caps. The algorithm for automatic creation of alignment zones respects

**Pro User Tip:** As opposed to the actual Filter UI, the Round Corner radius can be negative to round inner corners.

Adobe provides open source CMap Resources, including a sample Identity-0 ROS at [sourceforge.net/projects/cmap.adobe/files/](https://sourceforge.net/projects/cmap.adobe/files/)

this value. Also, when a small cap glyph is displayed in Edit view and metrics are set to show, the small cap height will be displayed instead of the x-height.

### 12.2.2 UFO3 Parameters

UFO3 Font Info properties  
are described on  
[unifiedfontobject.org/  
versions/ufo3/fontinfo.html](http://unifiedfontobject.org/versions/ufo3/fontinfo.html)

These custom parameters, again presented in alphabetical order, follow the naming convention for Font Info properties as set forth in the UFO3 specification published in March 2012. Text between quotes is taken from the OpenType specification on Microsoft's site. As of version 1.3.18, Glyphs also makes use of a simplified naming convention. Wherever possible, you can leave out the prefix of the keyword, e.g., instead of 'openTypeNameDescription', you can simply use 'description', or 'blueScale' instead of 'postscriptBlueScale'. Both long and short versions work side by side, though.

**blueScale** *float* BlueScale value. This corresponds to the Type 1/CFF BlueScale field. Controls the font size until which overshoot display is suppressed. Calculated as (pointsize at 300 dpi - 0.49) ÷ 240, e.g., 0.039625 for 10 points at 300 dpi. If you do not set the value yourself, blueScale defaults to 0.037, which corresponds to 9.37 points at 300 dpi or 39 pixels per em. This means that, in this case, overshoots will be visible if at least 40 pixels are used to display an em. The maximum blueScale value depends on the sizes of your alignment zones. The maximum pointsize at 300 dpi is calculated as  $0.49 + 240 \div \text{largest alignment zone size}$ , which corresponds to a PPM (size in pixels per em) of  $2.04 + 1000 \div \text{largest alignment zone size}$ . The product of (maximum pointsize - 0.49) × (largest alignment zone height) must be less than 240.

For example, your largest zone is 21 units deep, thus:  $2.04 + 1000 \div 21 = 49.659$ , so the maximum PPM at which overshoots can be suppressed is 49. The corresponding maximum pointsize is  $0.49 + 240 \div 21 = 11.919$  points at 300 dpi, thus the blueScale value cannot exceed  $(11.919 - 0.49) \div 240 = 0.04762$ .

**blueShift** *integer or float* BlueShift value. This corresponds to the Type 1/CFF BlueShift field. Default value is 7. Extends for very small glyph features beyond the font size indicated by blueScale. Overshoots inside an alignment zone are displayed if: (a) they are equal to or larger than BlueShift and (b) if they are smaller than BlueShift but larger than half a pixel. E.g. blueScale is set to suppress overshoots until 32PPM, blueShift is 6,

overshoots are 12 units deep. The stroke endings are slightly slanted and extend 5 units below the baseline. Between 0 and 32 PPM, the baseline will be kept completely level. Starting at 33 PPM, the overshoots will kick in. But the stroke endings will stay flat, because 5 units do not cover half a pixel until 100 PPM.

**compatibleFullName** *string* Compatible full name (Mac only).

Corresponds to the OpenType name table name ID 18. If not set, the value for name table ID 18 is calculated from Family Name plus space plus Style Name of the respective instance. 'On the Macintosh, the menu name is constructed using the FOND resource. This usually matches the Full Name. If you want the name of the font to appear differently than the Full Name, you can insert the Compatible Full Name in ID 18.'

**copyright** *string* Copyright statement. Overrides the entry in the Copyright field in the Font tab of the Font Info. Corresponds to the OpenType name table name ID 0.

**description** *string* Description of the font. Corresponds to the OpenType name table name ID 10: 'description of the typeface. Can contain revision information, usage recommendations, history, features, etc.'

**familyName** *string* Family name. Overrides the entry in the Family Name field in the Font section of the Font Info. Corresponds to the OpenType name table name IDs 1 and 4. Used to calculate IDs 3, 4 and 6.

**fsType** *list* A list of bit numbers indicating the embedding type. The bit numbers are listed in the OpenType OS/2 specification. Corresponds to the OpenType OS/2 table fsType field. 'Type flags. Indicates font embedding licensing rights for the font. Embeddable fonts may be stored in a document. When a document with embedded fonts is opened on a system that does not have the font installed (the remote system), the embedded font may be loaded for temporary (and in some cases, permanent) use on that system by an embedding-aware application. Embedding licensing rights are granted by the vendor of the font.'

The OpenType Font Embedding DLL Specification and DLL release notes describe the APIs used to implement support for OpenType font embedding and loading. Applications that implement support for font embedding, either through use of the Font Embedding DLL or through other means, must not embed fonts which are not licensed to permit embedding.

Please note that the embedding type is really just a usage suggestion for an application, not an actual protection mechanism. An application may ignore the fsType setting.

Further, applications loading embedded fonts for temporary use (see Preview & Print and Editable embedding below) must delete the fonts when the document containing the embedded font is closed.’ Possible settings:

*Not set.* ‘Fonts with this setting indicate that they may be embedded and permanently installed on the remote system by an application. The user of the remote system acquires the identical rights, obligations and licenses for that font as the original purchaser of the font, and is subject to the same end-user license agreement, copyright, design patent, and/or trademark as was the original purchaser.’

*Forbidden.* ‘Restricted License embedding: Fonts that have only this bit set must not be modified, embedded or exchanged in any manner without first obtaining permission of the legal owner. Caution: For Restricted License embedding to take effect, it must be the only level of embedding selected.’

*Preview & Print* ‘When this bit is set, the font may be embedded, and temporarily loaded on the remote system. Documents containing Preview & Print fonts must be opened ‘read-only;’ no edits can be applied to the document.’

*Editable.* ‘When this bit is set, the font may be embedded but must only be installed temporarily on other systems. In contrast to Preview & Print fonts, documents containing Editable fonts may be opened for reading, editing is permitted, and changes may be saved.’

*Subsetting forbidden.* ‘When this bit is set, the font may not be subsetted prior to embedding. Other embedding restrictions also apply.’

**hheaAscender** *integer* Ascender value. Corresponds to the OpenType hhea table Ascender field. ‘Typographic ascent (distance from baseline of highest ascender).’

**hheaDescender** *integer* Descender value. Corresponds to the OpenType hhea table Descender field. ‘Typographic descent (distance from baseline of lowest descender).’

**hheaLineGap** *integer* Line gap value. Corresponds to the OpenType hhea table LineGap field. ‘Typographic line gap. Negative LineGap values are treated as zero in Windows 3.1, System 6, and System 7.’

**italicAngle** *integer or float* Italic angle. This must be an angle in counter-clockwise degrees from the vertical. Overrides the entry in the Italic Angle field in the Masters tab of the Font Info.

**license** *string* License description. Corresponds to the OpenType name table name ID 13, the ‘description of how the font may be legally used, or different example scenarios for licensed use. This field should be written in plain language, not legalese.’

**licenseURL** *string* URL for the license. Corresponds to the OpenType name table name ID 14. ‘URL where additional licensing information can be found.’

**note** *string* Arbitrary note about the font.

**openTypeHheaAscender** *see* *hheaAscender*

**openTypeHheaDescender** *see* *hheaDescender*

**openTypeHheaLineGap** *see* *hheaLineGap*

**openTypeNameCompatibleFullName** *see* *compatibleFullName*

**openTypeNameDescription** *see* *description*

**openTypeNameLicense** *see* *license*

**openTypeNameLicenseURL** *see* *licenseURL*

**openTypeNamePreferredFamilyName** *see* *preferredFamilyName*

**openTypeNamePreferredSubfamilyName** *see*  
*preferredSubfamilyName*

**openTypeNameSampleText** *see* *sampleText*

**openTypeNameWWSFamilyName** *see* *WWSFamilyName*

**openTypeNameWWSSubfamilyName** *see* *WWSSubfamilyName*

**openTypeOS2Panose** *see* *panose*

**openTypeOS2Type** *see* *fsType*

**openTypeOS2TypoAscender** *see* *typoAscender*

**openTypeOS2TypoDescender** *see* *typoDescender*

**openTypeOS2TypoLineGap** *see* *typoLineGap*

**openTypeOS2UnicodeRanges** *see* *unicodeRanges*

**openTypeOS2VendorID** *see* *vendorID*

**openTypeOS2WeightClass** *see* *weightClass*

**openTypeOS2WidthClass** *see* *widthClass*

**openTypeOS2WinAscent** *see* *winAscent*

**openTypeOS2WinDescent** *see* *winDescent*

**openTypeVheaVertTypoAscender** *see* *vheaVertTypoAscender*

**openTypeVheaVertTypoDescender** *see* *vheaVertTypoDescender*

**openTypeVheaVertTypoLineGap** *see* *vheaVertTypoLineGap*

**panose** *list* The list must contain 10 non-negative integers that represent the setting for each category in the Panose specification. The integers correspond with the option numbers in each of the Panose categories. This corresponds to the OpenType OS/2 table Panose field. ‘This 10 byte series of numbers is used to describe the visual characteristics of a

More information about  
Monotype’s PANOSE  
classification system:  
[monotypeimaging.com/  
ProductsServices/pan1.aspx](http://monotypeimaging.com/ProductsServices/pan1.aspx)

given typeface. These characteristics are then used to associate the font with other fonts of similar appearance having different names. [...] The Panose values are fully described in the Panose “greybook” reference, currently owned by Monotype Imaging. The PANOSE definition contains ten digits each of which currently describes up to sixteen variations. Windows uses `bFamilyType`, `bSerifStyle` and `bProportion` in the font mapper to determine family type. It also uses `bProportion` to determine if the font is monospaced. If the font is a symbol font, the first byte of the PANOSE number (`bFamilyType`) must be set to “pictorial.”

**postscriptBlueScale** *see blueScale*

**postscriptBlueShift** *see blueShift*

**postscriptFontName** *string* Name to be used for the `FontName` field in Type1/CFF table. Should be ASCII-only, no whitespace allowed, e.g., ‘MyFont-Regular’.

**postscriptFullName** *string* Name to be used for the `FullName` field in Type1/CFF table.

**postscriptUnderlinePosition** *see underlinePosition*

**postscriptUnderlineThickness** *see underlineThickness*

**preferredFamilyName** *string* Preferred family name. Corresponds to the OpenType name table name ID 16. ‘For historical reasons, font families have contained a maximum of four styles, but font designers may group more than four fonts to a single family. The Preferred Family allows font designers to include the preferred family grouping which contains more than four fonts. This ID is only present if it is different from ID 1, the Family Name as set in Font Info.’

**preferredSubfamilyName** *string* Preferred subfamily name.

Corresponds to the OpenType name table name ID 17.

‘Preferred Subfamily; Allows font designers to include the preferred subfamily grouping that is more descriptive than ID 2. This ID is only present if it is different from ID 2 and must be unique for the Preferred Family.’

**sampleText** *string* Sample text. Corresponds to the OpenType name table name ID 19. ‘This can be the font name, or any other text that the designer thinks is the best sample to display the font in.’

**styleMapFamilyName** *string* Family name used for bold, italic, and bold italic style mapping. You can use this to create subfamilies within larger font families. ‘Up to four fonts can share the Font Family name, forming a font style linking group.’ Glyphs

uses the entries in Style Name field and in the Style Linking section in the Instances tab of the Font Info for linking the four individual weights.

**trademark** *string* Trademark statement. Corresponds to the OpenType name table name ID 7. According to Microsoft, ‘this is used to save any trademark notice / information for this font. Such information should be based on legal advice. This is distinctly separate from the copyright.’

**typoAscender** *integer* Ascender value. Corresponds to the OpenType OS/2 table sTypoAscender field. ‘The typographic ascender for this font. Remember that this is not the same as the Ascender value in the hhea table, which Apple defines in a far different manner. One good source for sTypoAscender in Latin based fonts is the Ascender value from an AFM file. For CJK fonts see below.

The suggested usage for sTypoAscender is that it be used in conjunction with unitsPerEm to compute a typographically correct default line spacing. The goal is to free applications from Macintosh or Windows-specific metrics which are constrained by backward compatibility requirements. These new metrics, when combined with the character design widths, will allow applications to lay out documents in a typographically correct and portable fashion. These metrics will be exposed through Windows APIs. Macintosh applications will need to access the sfnt resource and parse it to extract this data from the “OS/2” table.

For CJK (Chinese, Japanese, and Korean) fonts that are intended to be used for vertical writing (in addition to horizontal writing), the required value for sTypoAscender is that which describes the top of the of the ideographic em-box. For example, if the ideographic em-box of the font extends from coordinates 0, -120 to 1000, 880 (that is, a 1000 × 1000 box set 120 design units below the Latin baseline), then the value of sTypoAscender must be set to 880. Failing to adhere to these requirements will result in incorrect vertical layout.’

**typoDescender** *integer* Descender value. Corresponds to the OpenType OS/2 table sTypoDescender field. ‘The typographic descender for this font. Remember that this is not the same as the Descender value in the hhea table, which Apple defines in a far different manner. One good source for sTypoDescender in Latin based fonts is the Descender value from an AFM file. For CJK fonts see below.

The suggested usage for `sTypoDescender` is that it be used in conjunction with `unitsPerEm` to compute a typographically correct default line spacing. The goal is to free applications from Macintosh or Windows-specific metrics which are constrained by backward compatibility requirements. These new metrics, when combined with the character design widths, will allow applications to lay out documents in a typographically correct and portable fashion. These metrics will be exposed through Windows APIs. Macintosh applications will need to access the `sfnt` resource and parse it to extract this data from the “OS/2” table (unless Apple exposes the “OS/2” table through a new API).

For CJK (Chinese, Japanese, and Korean) fonts that are intended to be used for vertical writing (in addition to horizontal writing), the required value for `sTypoDescender` is that which describes the bottom of the of the ideographic em-box. For example, if the ideographic em-box of the font extends from coordinates 0,-120 to 1000,880 (that is, a 1000 × 1000 box set 120 design units below the Latin baseline), then the value of `sTypoDescender` must be set to -120. Failing to adhere to these requirements will result in incorrect vertical layout.’

**typoLineGap** *integer* Line gap value. Corresponds to the OpenType OS/2 table `sTypoLineGap` field. ‘The typographic line gap for this font. Remember that this is not the same as the `LineGap` value in the `hhea` table, which Apple defines in a far different manner.

The suggested usage for `usTypoLineGap` is that it be used in conjunction with `unitsPerEm` to compute a typographically correct default line spacing. Typical values average 7–10% of units per em. The goal is to free applications from Macintosh or Windows-specific metrics which are constrained by backward compatibility requirements. These new metrics, when combined with the character design widths, will allow applications to lay out documents in a typographically correct and portable fashion.’

**underlinePosition** *integer or float* Underline position value. Corresponds to the Type 1/CFF/post table `UnderlinePosition` field. Default is -100.

**underlineThickness** *integer or float* Underline thickness value. Corresponds to the Type 1/CFF/post table `UnderlineThickness` field. Default is 50.

Microsoft keeps a list of registered vendors at:  
[www.microsoft.com/typography/links/vendorlist.aspx](http://www.microsoft.com/typography/links/vendorlist.aspx)

**unicodeRanges** *list* A list of supported Unicode ranges in the font. Corresponds to the OpenType OS/2 table ulUnicodeRange1, ulUnicodeRange2, ulUnicodeRange3 and ulUnicodeRange4 fields. Glyphs offers a search field, so you can quickly spot the proper ranges for your fonts. E.g. if you want to cover all Latin ranges, simply search for 'latin' and all corresponding ranges will be filtered from the list.

**unitsPerEm** *non-negative integer* Units per em. Default is 1000 for PostScript-flavored OpenType fonts and a power of 2 (usually 2048) for TrueType-flavored OpenType fonts.

**vendorID** *string* Four character identifier for the creator of the font. Corresponds to the OpenType OS/2 table achVendID field. If not set, Glyphs will use 'UKWN' ('unknown') as Vendor ID. 'The four character identifier for the vendor of the given type face. This is not the royalty owner of the original artwork. This is the company responsible for the marketing and distribution of the typeface that is being classified. It is reasonable to assume that there will be 6 vendors of ITC Zapf Dingbats for use on desktop platforms in the near future (if not already). It is also likely that the vendors will have other inherent benefits in their fonts (more kern pairs, unregularized data, hand hinted, etc.). This identifier will allow for the correct vendor's type to be used over another, possibly inferior, font file. The Vendor ID value is not required.'

**vheaVertTypoAscender** *integer* Ascender value for vertical typesetting. Corresponds to the vertTypoAscender field in the OpenType vhea table.

**vheaVertTypoDescender** *integer* Descender value for vertical typesetting. Corresponds to the vertTypoDescender field in the OpenType vhea table.

**vheaVertTypoLineGap** *integer* Line gap value for vertical typesetting. Corresponds to the vertTypoLineGap field in the OpenType vhea table.

**weightClass** *integer* Weight class value. Must be a non-negative integer. Corresponds to the OpenType OS/2 table usWeightClass field. 'Indicates the visual weight (degree of blackness or thickness of strokes) of the characters in the font.'

Some applications use this value to sort the subfamilies in the font menu. Overrides the value set by the Weight pop-up menu of the instance.

<i>Value</i>	<i>Description</i>
100	Thin

200	Extra-light (Ultra-light)
300	Light
400	Normal (Regular)
500	Medium
600	Semi-bold (Demi-bold)
700	Bold
800	Extra-bold (Ultra-bold)
900	Black (Heavy)

**widthClass** *integer* Width class value. Must be in the range 1–9. Corresponds to the OpenType OS/2 table `usWidthClass` field. ‘Indicates a relative change from the normal aspect ratio (width to height ratio) as specified by a font designer for the glyphs in a font.

Although every character in a font may have a different numeric aspect ratio, each character in a font of normal width has a relative aspect ratio of one. When a new type style is created of a different width class (either by a font designer or by some automated means) the relative aspect ratio of the characters in the new font is some percentage greater or less than those same characters in the normal font — it is this difference that this parameter specifies.’

Some applications use this value to sort the subfamilies in the font menu. Overrides the value set by the Weight pop-up menu of the instance.

<i>Value</i>	<i>Description</i>	<i>% of normal</i>
1	Ultra-condensed	50
2	Extra-condensed	62.5
3	Condensed	75
4	Semi-condensed	87.5
5	Medium (normal)	100
6	Semi-expanded	112.5
7	Expanded	125
8	Extra-expanded	150
9	Ultra-expanded	200

The term ANSI refers to the Windows 8-bit encoding 1252, which is described at [msdn.microsoft.com/en-us/goglobal/cc305145](https://msdn.microsoft.com/en-us/goglobal/cc305145) and covers mostly Western European Latin characters.

**winAscent** *non-negative integer* Ascender value for Windows. Corresponds to the OpenType OS/2 table `usWinAscent` field. ‘`usWinAscent` is computed as the `yMax` for all characters in the Windows ANSI character set. `usWinAscent` is used to compute the Windows font height and default line spacing. For platform 3 encoding 0 fonts, it is the same as `yMax`. Windows will clip the bitmap of any portion of a glyph that

appears above this value.’ Thus, winAscent should be high enough to include caps and their accents.

**winDescent** *non-negative integer* Descender value for Windows. Corresponds to the OpenType OS/2 table usWinDescent field. ‘usWinDescent is computed as the -yMin for all characters in the Windows ANSI character set. usWinDescent is used to compute the Windows font height and default line spacing. For platform 3 encoding 0 fonts, it is the same as -yMin. Windows will clip the bitmap of any portion of a glyph that appears below this value.’ Thus, winDescent should be large enough to encompass the descenders of lowercase letters like g, p, q, and y.

**WWSFamilyName** *string* WWS family name. WWS stands for ‘Weight Width Slope’. Corresponds to the OpenType name table name ID 21. ‘Used to provide a WWS-conformant family name in case the entries for IDs 16 and 17 do not conform to the WWS model. (That is, in case the entry for ID 17 includes qualifiers for some attribute other than weight, width or slope.) [...] Examples of name ID 21: “Minion Pro Caption” and “Minion Pro Display”. (Name ID 16 would be “Minion Pro” for these examples.)’

**WWSSubfamilyName** *string* WWS Subfamily name. Corresponds to the OpenType name table name ID 22. ‘Used in conjunction with ID 21, this ID provides a WWS-conformant subfamily name (reflecting only weight, width and slope attributes) in case the entries for IDs 16 and 17 do not conform to the WWS model. [...] Examples of name ID 22: “Semibold Italic”, “Bold Condensed”. (Name ID 17 could be “Semibold Italic Caption”, or “Bold Condensed Display”, for example.)’ For name IDs 16 and 17, see the entries for preferredFamilyName and preferredSubfamilyName, respectively.