Using Glyph Transformations

In High-Logic® FontCreator Professional 12.0

The transform feature is a powerful feature for make new glyphs or font styles from existing ones. It is not available in the Home Edition, but only in the Standard or Professional Editions, or in the thirty-day Free Trial. Please see this » Comparison Chart « and the » Registration Page « for details.

FontCreator 12.0 brings some major changes to Glyph Transformations: Transformations can now be undone, code-points in the Private Use Area are no longer used (except for a few Nordic glyphs as proposed by the Medieval Unicode Font Initiative), and it supports Anchored Based generation of composites.

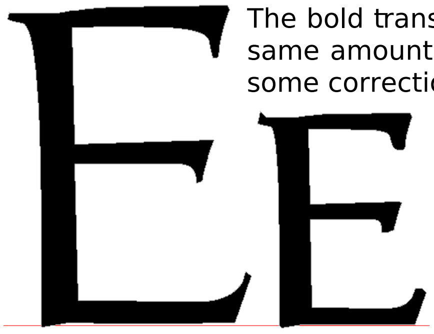
General Advice for Working With Transformations

Select one or more glyphs in the glyph overview, then from the Tools menu, select Glyph Transformer. Click the folder icon to open one of several predefined scripts. User-defined scripts can be created by adding any of the available features from the left panel, and saving them for later reuse. If run in a glyph edit window the script will affect only the current glyph.

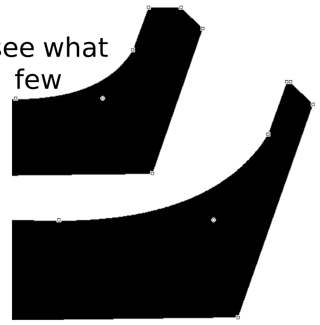
Small Capital Fonts

Before running this transformation, copy the Capital Letters of the font to the lowercase positions, and select them. The Small Capitals transformation scales the uppercase by about 75%. To compensate for the scaling the stroke weight is increased with a bold transformation. To add Small Capitals for OpenType Features use the Unmapped Petite or Small Capitals scripts. There are scripts for Latin, Greek, and Cyrillic.

Script: Small Capitals.xml
Scale (75.00, 78.00) (0, 0)
Bold (20,12) preserve bearings
Move (0,12)
Optimize

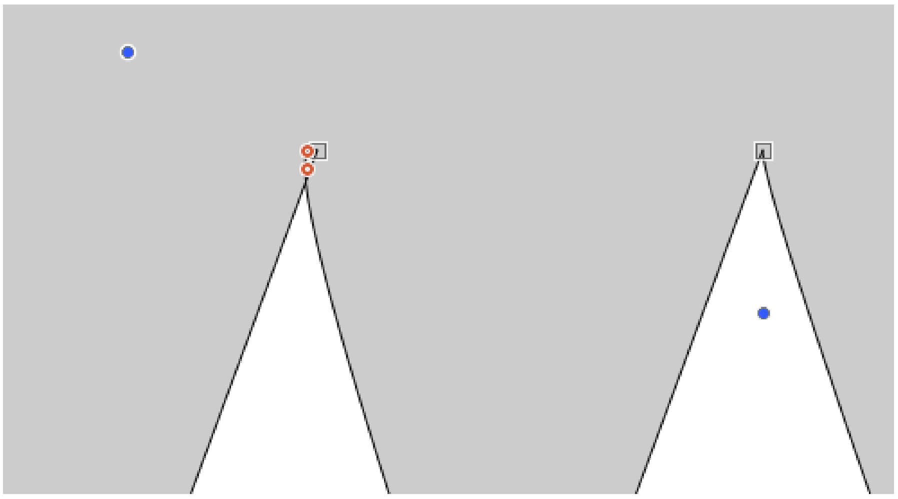


The bold transformation is imperfect. Thin strokes and serifs are made thicker by the same amount as thick strokes. After applying the transformation one needs to make some corrections. This is easy when one knows what to look for. Compare the original E and the transformed result.



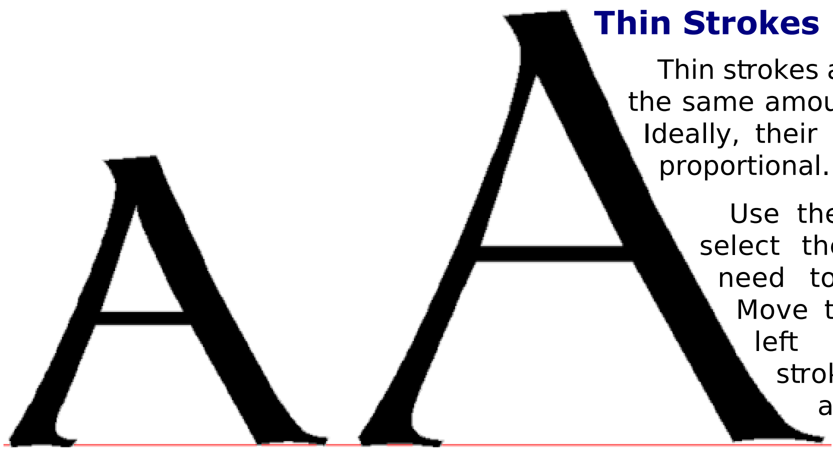
The Serifs Are Too Thick — Zoom in close to see what has happened. All one needs to do is select a few nodes with the lasso selection tool and move them to the right using the cursor keys. The serifs have been made 40 funits thicker when they only need to increase by about 10 funits.

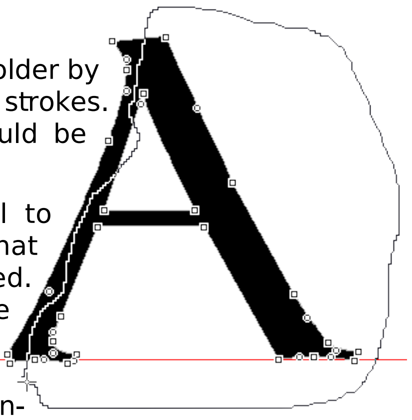
Intersecting Co-ordinates — Sometimes a node crosses the contour giving a result like that shown below (Capital A). Just zoom in close and move or delete the offending node to correct this problem.



Contents

Thin Strokes





Thin strokes are made bolder by the same amount as thick strokes. Ideally, their weight should be proportional.

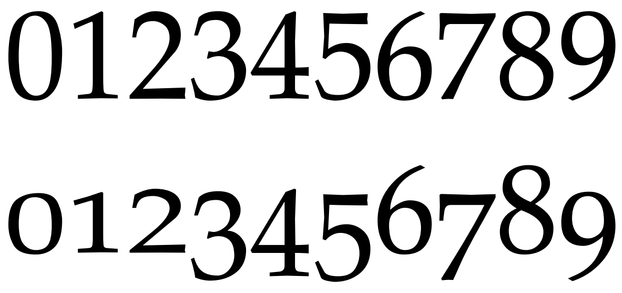
Use the lasso tool to select the nodes that need to be moved. Move them to the left until the stroke weights are proportional. Reduce the advance width to match.

OldStyle Figures

Old style figures or non-lining figures are the default figures in some fonts like Georgia or Constantia, but most fonts have lining figures, which are designed for use with Capitals. Non-lining figures are designed for use with body text with the digits being designed on the x-height.

Fonts that have Old Style Figures by default may need lining figures for use with capitals. A similar script could be used to insert lining figures scaling up the small figures, and moving the figures with descenders up to the baseline.

Script: OldStyle Figures.xml
Insert Glyphs zero.onum-two.onum
Complete Composites
Decompose Composites
Insert Glyphs three.onum-five.onum
Move (0,-393)
Insert Glyphs six.onum
Complete Composites
Insert Glyphs seven.onum
Move (0,-393)
Insert Glyphs eight.onum
Complete Composites
Insert Glyphs nine.onum
Complete Composites
Move (0,-393)



This script inserts Unmapped Old Style Figures. It first inserts the figures 0, 1, and 2, and Complete Composites scales them by 75% vertically to reduce them (approximately) to the x-height. Then it inserts the figures 3, 4, and 5, moving them vertically by the difference between CapHeight and x-height. Figures six and eight do not need moving down, while seven and nine do, so each is processed in numerical order. The script could be shorter, but then the glyphs will not be in order.

Glyphs 0, 1 and 2 will need to be edited, that is why they are decomposed.

The figure 4 may also need decomposing editing to remove its “foot” since it hangs below the baseline rather than standing on it as in lining figures. The other figures — 3, 5, 7, 8, and 9 — have the same design as lining figures, but are positioned and/or spaced differently.

Small and Petite Capitals

Small Capitals may replace the lowercase letters in a font to create a dedicated Small Caps font, or they may use the OpenType Small Capitals feature with glyph substitutions to replace lowercase letters. The small capitals are similar in proportion to the uppercase, and typically 80% of the height. The stroke weights will need to be increased to compensate for the scaling.

Petite Capitals are designed to be substituted for lowercase letters when the Petite Capitals attribute is applied in applications. They may have a wider aspect ratio than uppercase and should match the x-height of the lowercase. In OpenType fonts they are accessed by using glyph substitutions.

Typographical variants like Small Capitals are not included in the Unicode standard. There are some small capitals in the Unicode charts, but they are IPA phonetic glyphs, not typographical variants.

Script: Unmapped Latin Small Capitals.xml
Insert Glyphs exclam.smcp, dollar.smcp, percent.smcp, ampersand.smcp, parenleft.smcp, parenright.smcp, question.smcp, bracketleft.smcp, bracketright.smcp, a.smcp, b.smcp, c.smcp, d.smcp, e.smcp, f.smcp, g.smcp, h.smcp, i.smcp, j.smcp, k.smcp, l.smcp, m.smcp, n.smcp, o.smcp, p.smcp, q.smcp, r.smcp, s.smcp, t.smcp, u.smcp, v.smcp, w.smcp, x.smcp, y.smcp, z.smcp, braceleft.smcp, braceright.smcp, exclamdown.smcp, sterling.smcp, yen.smcp, thorn.smcp, eng.smcp, Germandbls.smcp, questiondown.smcp
Complete Composites
Decompose Composites
Scale (78.00, 82.00) (0,0)
Bold (16,12)
Move (0,12)
Insert Glyphs germandbls.smcp agrave.smcp aacute.smcp acircumflex.smcp atilde.smcp adieresis.smcp aring.smcp ae.smcp ccedilla.smcp egrave.smcp eacute.smcp ecircumflex.smcp edieresis.smcp igrave.smcp iacute.smcp icircumflex.smcp idieresis.smcp eth.smcp ntilde.smcp ograve.smcp oacute.smcp ocircumflex.smcp otilde.smcp odieresis.smcp oslash.smcp ugrave.smcp uacute.smcp ucircumflex.smcp udieresis.smcp yacute.smcp ydieresis.smcp amacron.smcp abreve.smcp aogonek.smcp cacute.smcp ccircumflex.smcp cdotaccent.smcp ccaron.smcp dcaron.smcp dcroat.smcp emacron.smcp ebreve.smcp edotaccent.smcp eogonek.smcp ecaron.smcp gcircumflex.smcp gbreve.smcp gdotaccent.smcp gcedilla.smcp hcircumflex.smcp hbar.smcp itilde.smcp imacron.smcp ibreve.smcp iogonek.smcp idotaccent.smcp ij.smcp jcircumflex.smcp kcedilla.smcp lacute.smcp lcedilla.smcp lcaron.smcp lmiddledot.smcp lslash.smcp nacute.smcp ncedilla.smcp ncaron.smcp omacron.smcp obreve.smcp odoubleacute.smcp oe.smcp racute.smcp rcedilla.smcp rcaron.smcp sacute.smcp scircumflex.smcp scedilla.smcp scaron.smcp tcedilla.smcp tcaron.smcp tbar.smcp utilde.smcp umacron.smcp ubreve.smcp uring.smcp udoubleacute.smcp uogonek.smcp wcircumflex.smcp ycircumflex.smcp zacute.smcp zdotaccent.smcp zcaron.smcp gacute.smcp aringacute.smcp aeacute.smcp oslashacute.smcp scommaaccent.smcp tcommaaccent.smcp ddotbelow.smcp hdotbelow.smcp ldotbelow.smcp ldotbelowmacron.smcp mdotaccent.smcp mdotbelow.smcp ndotaccent.smcp ndotbelow.smcp rdotbelow.smcp rdotbelowmacron.smcp sdotbelow.smcp tdotbelow.smcp wgrave.smcp wacute.smcp wdieresis.smcp ygrave.smcp ytilde.smcp
Complete Composites

The script first inserts the base glyphs, then scales them, makes them bolder, and moves them up to compensate for the vertical scaling. The composite glyphs are then inserted and composed from diacritics that are available in the font. It is recommended to ad smaller case-sensitive (*.case) and/or narrow (*.narrow) combining diacritics to use with Petite Capitals or Small Capitals to suit the smaller glyphs.

The script for Petite Capitals is very similar, but with a different scale factor and slightly more horizontal bolding. My aim in designing the scripts was to match the stroke weight of lowercase glyphs after scaling to the x-height. They also have a slightly squarer ascpect ratio so that applying Petite Capitals to some text does not substantially alter copy-fit. This methodology may not suit all fonts.

Titling Capitals

Titling capitals are designed for use at larger point sizes, where regular capitals are too heavy. The glyphs retain the vertical height of regular capitals, but reduce the stem weight.

Script: Unmapped Latin Titling Capitals.xml
Insert Glyphs A.titl, B.titl, C.titl, D.titl, E.titl, F.titl, G.titl, H.titl, I.titl, J.titl, K.titl, L.titl, M.titl, N.titl, O.titl, P.titl, Q.titl, R.titl, S.titl, T.titl, U.titl, V.titl, W.titl, X.titl, Y.titl, Z.titl
Complete Composites
Decompose
Scale (95.00, 101.00) (0, 0)
Thin 7, 8
Move 0, -8

This script inserts Titling Capitals composed from capitals A-Z and reduces the stroke weight. Serifs, or any other thin strokes, will need adjusting afterwards if they are too thin. Adjust the scaling, thin, and move values to suit the particular needs of your font, and save the script for reuse. Undo the transformation and try with different values until you get the desired results.

CompositeData.xml currently only supports A-Z for Titling Capitals.

Italic and Oblique

The italic transformation works well, though it can only make an oblique version of a font, not a true italic, which requires cursive lowercase letters. Uppercase letters and figures usually look fine, though a few require different designs. Maths symbols can be slanted or upright in italic fonts. There is no hard and fast rule. Other glyphs such as the notdef glyph, vertical line © ¤ ± × ® ™ and geometric shapes will also look better if not slanted, though they may need moving to the right.

Before running this transformation, if the font is a hand-written script, set the Family Kind in Font, Properties, Characteristics to Latin Hand Written. Otherwise, it will be assumed to be Latin Text. When satisfied with the results, choose Save As... from the file menu, and give the italic font a new filename. The font's style is only changed to italic by the script if the option "Set font subfamily and font design to italic" is checked. Otherwise, though the glyphs are slanted, it remains a regular type style. This is what is usually required with a hand-written script.

Script: Italic.xml
Italic 11.00 deg, update
Left Side Bearing Point at x=0
Optimize

After skewing, the glyphs will have a lot of off-curve extremes. These are removed automatically by the feature "Optimize." Other errors may need to be fixed manually.

Eastern Europe, Greek Extended, and Vietnamese

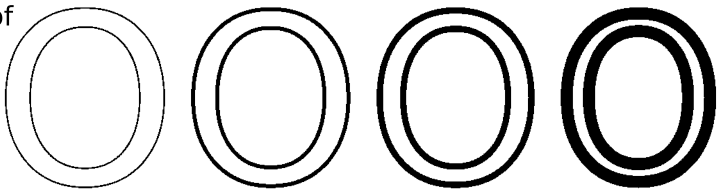
These scripts insert a wide range of extended characters. Vietnamese requires characters in the Latin Extended Additional range and stacking diacritical marks. Some work will be needed to edit the accents and align them correctly in composites. The » Gentium font « was used to design the Greek script.

Eastern Europe.xml, Basic Greek.xml, Extended Greek.xml, and Vietnamese.xml

These all use the Insert Characters, Complete Composites, and Decompose commands. After running the scripts, edit the diacritics for size, weight, and vertical position to suit the typeface. Then select composites, decompose them, and complete composites again.

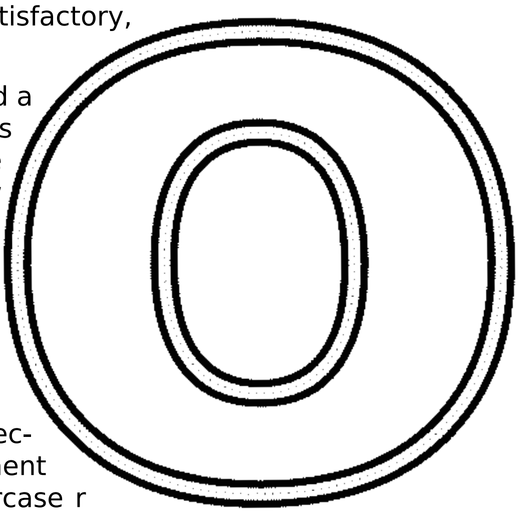
Outline or Hollow

Select all of the glyphs to transform, open one of the Outline scripts: Light, Medium, Bold, or Heavy.



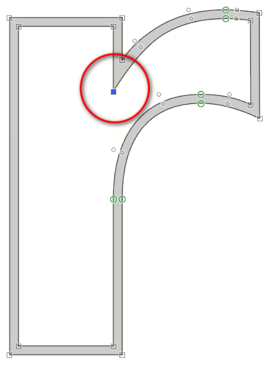
Script: Outline Medium.xml
Scale (97.00, 98.00) (0, 0) preserve bearings
Hollow (30, 30) preserve bearings
Move (0, 15)
Left Side Bearing Point at x=0

First, select the glyphs to be transformed. If the result is unsatisfactory, undo the transformation and adjust it.



The hollow transform enlarges the glyph in each direction, and a reversed (white) contour is generated inside it. The dotted lines shows the original contours of the glyph. The outline scripts scale the glyph down to compensate for the size increase from the hollow transformation, while maintaining the advance width.

Problem Glyphs



As with many transformations, there may be some glyphs where it does not work perfectly.

Wherever there is a sharp change of direction in the contours, some manual adjustment may be needed afterwards as in this lowercase r (illustrated).

Switch to Points Mode, select one or more nodes with the mouse or w/q and shift+w/q shortcuts then move them with the cursor keys

Maths Symbols and Miscellaneous Symbols

These two scripts set the advance width and centre the glyph. In most fonts the maths symbols like < = > + ± should be the same width as the figures. In my Verajja font I also match the vertical arrows to the figure width. Modify the width value to suit the current font and save the transformation for reuse.

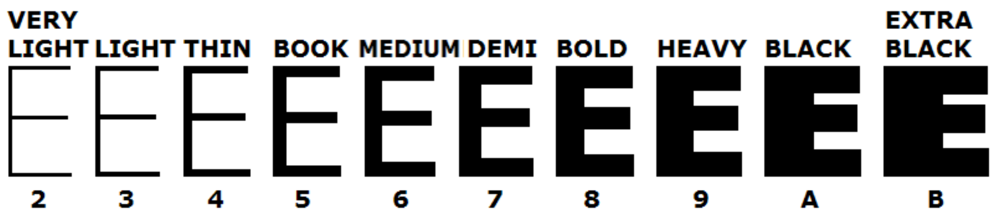
Script: Maths Symbols.xml
Override Range by Glyph Name dollar plus zero one two three four five six seven eight nine less equal greater cent sterling currency yen logicalnot plusminus mu paragraph multiply divide figurespace figuredash referencemark overline reversedpilcrowsign swungdash dottedcross colonsign cruzeirosign frenchfrancsign lirasign nairasign dongsign euro kipsign tugriksign germanpennysign pesosign hryvniasign cedisign livretournoissign spesmilosign tengesign indianrupeesign turkishlirasign nordicmarksign rublesign larisign bitcoinsign upwardsarrow downwardsarrow updownarrow upwardstwoheadedarrow downwardstwoheadedarrow upwardsarrowfrombar downwardsarrowfrombar updownarrowwithbase upwardsarrowwithtipleftwards upwardsarrowwithtiprightwards downwardsarrowwithtipleftwards downwardsarrowwithtiprightwards rightwardsarrowwithcornerdownwards upwardsharpoonwithbarbrightwards upwardsharpoonwithbarbleftwards downwardsharpoonwithbarbrightwards downwardsharpoonwithbarbleftwards upwardsarrowleftwardsofdownwardsarrow upwardspairedarrows downwardspairedarrows upwardsdoublearrow downwardsdoublearrow updowndoublearrow upwardsarrowwithdoublestroke downwardsarrowwithdoublestroke upwardsdashedarrow downwardsdashedarrow upwardswhitearrow downwardswhitearrow upwardswhitearrowfrombar upwardswhitearrowonpedestal upwardswhitearrowonpedestalwithhorizontalbar upwardswhitearrowonpedestalwithverticalbar upwardswhitedoublearrow upwardswhitedoublearrowonpedestal updownwhitearrow downwardsarrowleftwardsofupwardsarrow complement elementof notanelementof containsasmember doesnotcontainasmember endofproof minus minusorplus dotplus setminus squareroot cuberoot fourthroot proportionalto infinity logicaland logicalor intersection union therefore because proportion tildeoperator minustilde asymptoticallyequalto approximatelyequalto almostequalto tripletilde notequalto identicalto lessthanorequalto greaterthanorequalto subsetof supersetof
Width (fixed 1303) Both Sides
Center Glyph
Left Side Bearing Point at x=0

The Miscellaneous Symbols transformation is intended for use with Miscellaneous Symbols, Dingbats, Geometric Shapes, Arrows, etc. Many of these should be the same width. Use the icon on the Overview Toolbar to group glyphs by Advance width to identify those that are nearly the same width within a range of 10 funits. Copy selected glyphs and paste them into the transform script to change the range of glyphs selected for transformation. It is much more efficient in the long run as symbols may be shared between many fonts. The script as supplied with FontCreator selects two groups of glyphs and sets them to two different advance widths. Edit the glyph lists and adjust the width values to suit your needs.

Condensed Thin

This transformation scales the glyph horizontally, reduces the stroke weight, and scales the glyph vertically to compensate for the resulting reduction in the glyph height. It was designed to maintain the glyph height, and to reduce the vertical stroke weight from 202 funits to 82. This results in a Panose Classification of “Thin.” See Dave Crosby’s » Panose Tutorial « for an explanation of the Weight Ratio.

Weight	Ratio
V Light	≥ 35
Light	≤ 18 < 35
Thin	≤ 10 < 18
Book	≤ 7.5 < 10
Medium	≤ 5.5 < 7.5
Demi	≤ 4.5 < 5.5
Bold	≤ 3.5 < 4.5
Heavy	≤ 2.5 < 3.5
Black	≤ 2.0 < 2.5
X Black	< 2



Different fonts need different values for scaling and for reducing the stroke weight. FontCreator allows scaling to an accuracy of two decimal places.

Script: Condensed Thin.xml
Scale (90.00, 104.93) (0, 0)
Thin (40, 37) preserve bearings
Move (0, -37)
Optimize
Left Side Bearing Point at x=0

As with the previous transformations, undo and try different values until the ideal result is achieved. Some editing of points may be needed to adjust the weight of serifs and other thin strokes.

Bold Transformations

The previous illustration shows the full range of Panose weights from Very Light to Extra Black. When making a bold style from a regular style one needs to consider just how much bolder to make it. To calculate the Panose weight, measure the vertical stem of the uppercase E, and its vertical height. Divide the latter by the former to get the Weight Ratio. Verajja Regular has a capital height of 1493 funits and a vertical stem width of 202 funits, its Panose weight ratio is 7.39, so it is of medium weight, but at the thin end of the range.

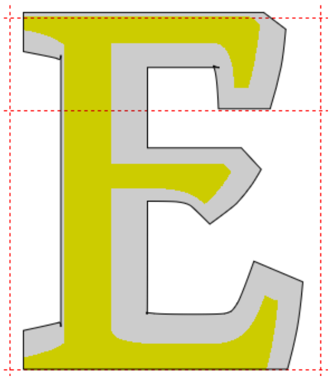
To make a bold style from the regular, open the Medium to Bold transform script and run it on the capital E to see if the result is what is wanted. This gives a stem weight of 350 and a capital height of 1494, so the weight ratio is 4.28, which is approaching the Demi Bold end of the bold range (≤ 3.5 < 4.5).

Script: Medium to Bold.xml
Scale (92.40, 92.40) (0, 0) preserve bearings
Bold (81, 57) preserve bearings
Move (0, 57)
Optimize
Left Side Bearing Point at x=0

Verajja Bold has a capital height of 1493 funits and a vertical stem width of 385 funits, its Panose weight ratio is 3.88, so it is of bold weight, towards the heavy end of the range. To make a heavy style from the bold, run the Bold to Heavy transform script. This results in a stem weight of 509 and a capital height of 1493, so the weight ratio is 2.93, which is near the middle of the Heavy range.

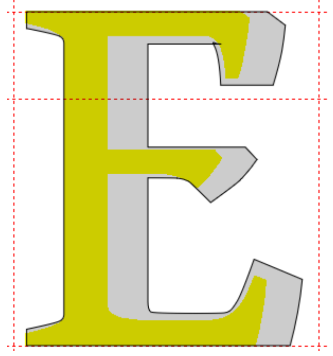
The scripts were tested on a font with even contrast and straight lines. When used on fonts with complex curves and sharply contrasting strokes the results may be far from ideal. Do not give up at once. By modifying the script the results can be improved. If that is still not good enough, the thick and thin strokes may have to be separated, using the techniques described in the Thin With High Contrast tutorial.

Correcting Bold Transformations



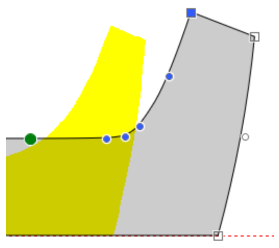
The Medium to Bold script was tested on Gentium Plus, which has no bold type style. The contrast between thick and thin strokes is low, so that should not cause too many problems. However, the vertical/horizontal contrast is greater than for Verajja, so the script will need to be modified. Copy the background image from the current glyph using the Background Image Toolbar (F9) before running the script to compare the before and after results.

The initial results are disappointing — the vertical stroke is correct, but the horizontal strokes and the serifs are much too heavy. The capital height has also increased, whilst it should be the same as the regular type style. The vertical bold effect needs reducing. A vertical bold value of 17 will add 34 funits to the vertical height After scaling the height should be 1260 – 34 = 1226. The scale factor should therefore be 1226/1260 * 100 = 97.30%



Script: Medium to Bold High Contrast.xml
Scale (97.30, 97.30) (0, 0) preserve bearings
Bold (81, 17) preserve bearings
Move (0, 17)
Left Side Bearing Point at x=0

The horizontal strokes are now right, the capital height is right, and the vertical strokes are right. The weight ratio is 3.97, which is in the middle of the bold range. Adjusting the horizontal bold effect is easy if the font is too bold or too heavy. The only problem is the serifs. Unless the serifs are cut off and transformed separately, the nodes will have to be adjusted. Save the script with a new name, undo the glyph transformation and apply it to the full alphabet.

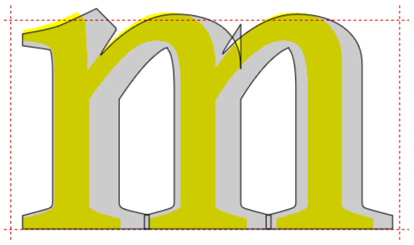


Fonts with fewer nodes are easier to edit. Optimize the contours to reduce the number of nodes before starting. Adjust the serifs by selecting multiple nodes and using the cursor keys to move them.

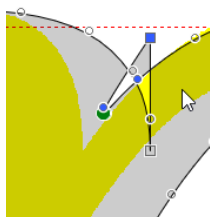
Having carefully adjusted the serifs on one glyph, cut and paste them to similar glyphs. Use the knife tool to cut the serifs off, copy them to the clipboard, and undo before pasting them into other glyphs. Then go back and weld them on with “Get Union of Contours.”

Bold Transformations of Lowercase

In most fonts the stems of the lowercase are thinner than those of the uppercase. If the same transformation is applied to the whole font, the lowercase, numerals, punctuation, and currency symbols will be too bold. Adjust the script to suit the lowercase. The lowercase “l” is a good choice on which to base the script, but check that the stem width matches other lowercase glyphs. For Gentium it is 150 funits, compared to 160 funits for uppercase “E.” The horizontal strokes are the same, so the script just needs slightly less bold width. Since the ascenders are taller than the capitals, the scale factor also needs adjusting slightly.



Script: Medium to Bold Lowercase.xml
Scale (96.10, 96.10) (0, 0) preserve bearings
Bold (76, 17) preserve bearings
Move (0, 17)
Optimize
Left Side Bearing Point at x=0



Some glyphs don’t need much work, but others seem to be beyond salvage. Fix these awkward glyphs by comparing them with the original, then use the lasso tool or Shift+Q/W shortcuts to select adjacent nodes. Bold and Regular styles should have a similar x-height. When all editing is finished, restore the right side bearing of the original glyph to correct the advance width.

Do not duplicate any detailed editing work. Copying the “m” that had already been edited, delete all of the nodes in the right leg, and adjusting the right side bearing took about two minutes instead of twenty. More importantly, the glyphs are the same shape and proportion. Before fixing the whole font it is best to preview the results. After running the script on all lowercase glyphs, check how they look in the Preview Toolbar (F8). Since the bold is too heavy, it would be better to use the Book to DemiBold script, which doesn’t increase the weight by quite so much.

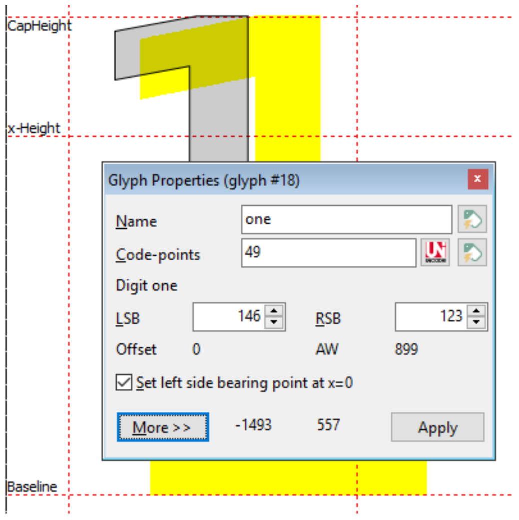
abcdefghijklmnopqrstuvwxyz

Making a bold style from a regular font is more difficult than one might think. That may be why Victor Gaultney has not yet released » a bold version of Gentium after many years.

Superscripts

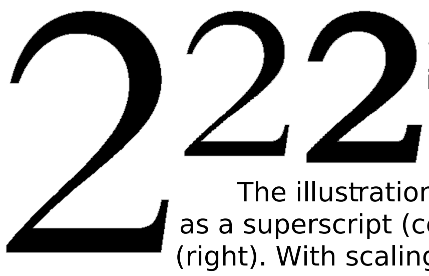
If superscripts are simply scaled from the existing figures in a font, they will be too light compared to the rest of the font.

First, measure the stem weight of digit 1. In this font it is 160 funits. If the superscript transformation scales this by 65% horizontally it will be only 104 funits. To compensate, one needs to use the bold transform with a horizontal value of 28 funits. This will thicken the stroke by 28 funits in both directions, resulting in a vertical stroke of 160 funits: 104 + 28 + 28.



Glyph Properties (glyph #18)

Name: one
Code-points: 49
Digit one
LSB: 146 RSB: 123
Offset: 0 AW: 899
☑ Set left side bearing point at x=0
More >> -1493 557 Apply



The horizontal strokes are 140 funits. If the vertical scale factor is 60% the vertical transformation would again need to be 28 (140 - 84)/2

The illustration shows the digit 2, the digit 2 as a superscript (centre), and a true superscript ^{2} (right). With scaling alone, the superscript is too light. The weight of the superscript is correct after the bold transformation has been applied. However, the thin strokes and serifs are now too heavy. They need to be adjusted by moving a few nodes.

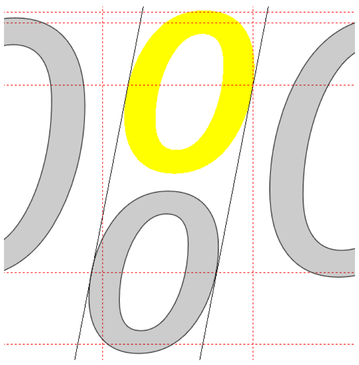
Each font will need a different transformation. The transform script just uses values to suit Verajja. The script aligns the tops of superscripts with the tops of numerals (1^1). Some designers may prefer to raise superscripts higher. If a font already contains superscripts ^{123} the script will only add the missing ones.

Script: Superscript.xml
Insert Characters 8304, 185, 178, 179, 8308-8316 (Apply subsequent features)
Complete Composites, Decompose
Scale (65.00, 60.00) LT
Bold (26, 20) preserve bearings
Move (0, -20)
Width (fixed 899) both sides (Remove this line if the figures are not of uniform width)
Left Side Bearing Point at x=0
Insert Characters 8305,8317,8318,8319
Complete Composites, Decompose
Scale (65.00, 60.00) LT
Bold (26, 20) preserve bearings
Move (0, -20)
Left Side Bearing Point at x=0

Superscript width (899) = figure width (1303) x scale (65/100) + Bold (2 x 26)

Subscripts

After adding the superscripts, use the Subscript Transformation to insert the Subscripts. This merely creates composites from the superscripts, and the definition in CompositeData.xml moves them down to bisect the baseline.



Script: Subscript.xml
Insert Characters 8320-8334 (Apply subsequent features)
Complete Composites

Superscripts and Subscripts in Italic type styles need to be offset horizontally to compensate for the italic angle. The recommended position is centred between figure zeros. A convenient way to do this is by using the Comparison Toolbar (F11) and move glyphs with the left/right cursor keys.

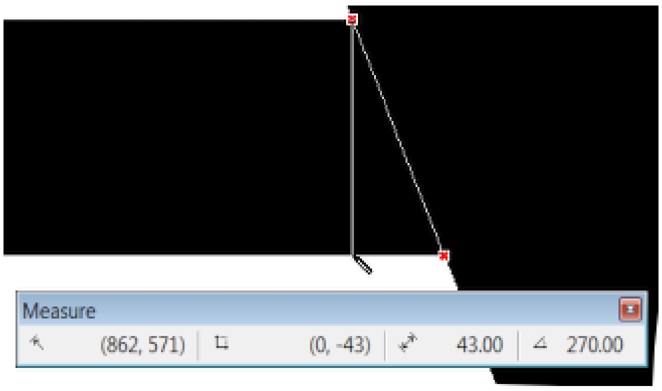
The default transform scripts do not insert all possible superscripts and subscripts, only the figures, basic maths symbols, and parentheses. I also add super/subscripts for × and ÷ for use with fractions.

Some applications won't use super/subscripts even if they exist in a font. They will just scale the numerals, which doesn't give the best results. Again, not all applications use the super/subscript data on the Font, Properties, General as this data is often missing or wrong. The situation will only improve if more font designers take the trouble to design their fonts properly.

Thin With High Contrast

If the glyphs have even strokes one can get a satisfactory result with a single transformation. However, if the glyphs have a high contrast between thick and thin strokes one must apply the transformation several times to the thick strokes, but only once to the thin strokes. Bold and heavy transforms on fonts with high contrast can use a similar method.

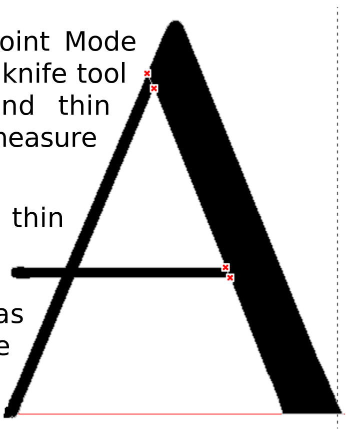
Cutting the Glyph



(862, 571) (0, -43) 43.00 270.00

Zoom in close using Point Mode and cut the glyph with the knife tool to separate the thick and thin strokes. At the same time, measure the thin and thick strokes.

For this glyph, the thin strokes were 43 funits and the thick strokes were 184 funits. The script was designed to reduce the stroke weights to 33 funits and 144 funits respectively.



Applying the Transformation

After cutting, the glyph looks like this in Contour Mode. Cut the thin strokes to the clipboard and open the Glyph Transform Wizard dialogue.

Clear any existing transformations by clicking on the delete icon. From the Effects group on the left, choose the Thin Transformation and click the right arrow to add it to the script. Enter the value for the horizontal transformation. Check the box to preserve side bearings. From the Metrics group on the left choose “Left Side Bearing Point at x=0” and add it to the script.

Thin (5, 0) preserve bearings
Left Side Bearing Point at x=0

Click OK to apply the transformation once to the remaining thick stroke. Repeat twice more, then paste the thin strokes back, and apply the transformation again. Total, four times for the thick stroke, and once for the thin strokes. To make the horizontal strokes thinner too, do this at the final stage with:

Thin (5, 10) preserve bearings
Move (0, -10)
Left Side Bearing Point at x=0

Zoom in and align the strokes using Control + Left/Right Cursor keys so that they join neatly at the top. Select all and join the contours together with "Get Union of Contours" from the Glyph Toolbar. Delete any unwanted nodes to complete the glyph.

Discretionary Ligatures

In the days of metal type, many letter pairs were created as ligatures. Only a few of these have survived in most modern fonts — ff, fi, fl, ffi, ffl, and occasionally one finds ligatures with long s (ſt), st and ct. The latter doesn’t have a code-point, but the former pairs are encoded in the Alphabetic Presentation Forms character set. A few fonts include historical ligatures found in the Latin Extended-D character set, and others contain discretionary ligatures for purely decorative purposes.

AA aa AO ao AU au AV av AY ay OO oo VY vy sä sh si sl sö sp ss ssi ssl ſti str sü sb sk (hlig) ck ct fj fr ft fy fft ffy fty sp tr tt tty ty tz Qu Th ttr FF FI FL HE LA MB MD ME MP MR NK NT OC OG OO TT TW TY UB UD UL UP UR cky ky ffr ip it tw fb ffb fh ffh fk ffk ffj fij íj (dlig)

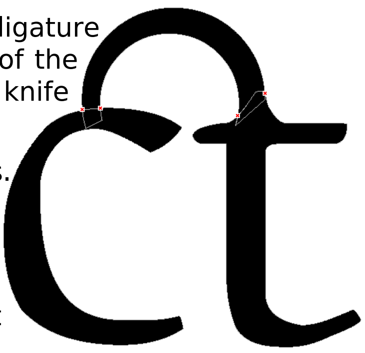
This script inserts the above characters. If a font doesn’t yet include the standard ff, fi, fl ligatures, it will insert them too. After running the script, connecting contours may be needed. There is scope to be creative with discretionary ligatures, but standard ligatures in the Alphabetic Presentation Forms (ff - ffl) should closely match the standard letter forms because standard ligatures are usually enabled by default.

ff fi fl ffi ffl fff

ff fi fl ffi ffl fff

Script: Unmapped Discretionary Ligatures.xml
Insert Glyphs c_k c_t f_j f_r f_t f_y f_f_t f_f_y f_t_y s_p t_r t_t t_t_y t_y Q_u T_h t_t_r F_F F_I F_L H_E L_A M_B M_D M_E M_P M_R N_K N_T O_C O_G OO T_T T_W T_Y U_B U_D U_L U_P U_R c_k_y k_y f_f_r i_p i_t t_w f_b f_f_b f_h f_f_h f_k f_f_k f_f_j f_i_j iacute_j_acutecomb
Complete Composites

Joining Ligatures

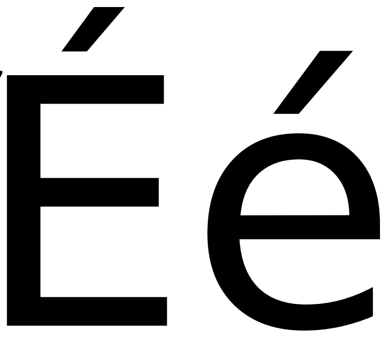


Some ligature pairs need to be joined with new contours. To create this ct ligature a ring from the samples toolbar (F12), was resized to suit, then the centre of the ring was offset to make it thinner at bottom left. After cutting the ring with the knife tool, the three contours are ready to be joined with “Get Union of Contours.”

The same connecting contour can be used for the st and sp ligatures. Reusing contours helps to maintain a consistent design. The copy can be modified to fit the new pair, but the weight of the stroke especially should be the same for all related glyphs.

Many Alphabetic Presentation Forms need to be kerned, but be careful not to overdo it. The spacing between the stems of f i and l should be uniform:

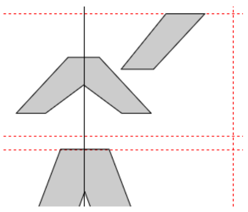
Uppercase Diacritics



Some fonts like Bitstream Vera (on which Verajja is based) use smaller accents for uppercase letters than they do for lowercase. If they exist, Complete Composites will use them for uppercase. Those with the ".case" suffix are for use with uppercase base glyph; those with the ".narrow" suffix are for use with narrow glyphs: I, i, l, J, j. Those with the ".cap" suffix are for use below Petite and Small Capitals. The .case accents should be positioned vertically for use with uppercase while the .narrow accents should be positioned vertically for use with lowercase. This script will insert unmapped diacritics and compose them from regular accents. That is the first step in the design process. Their height needs to be reduced to suit the design of the font, while maintaining their weight so that they match other accents.

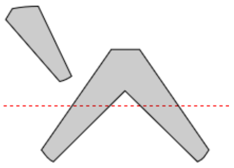
Script: Unmapped Uppercase Diacritics.xml
Insert Glyphs gravecomb.case acutecomb.case circumflexcomb.case circumflexcomb.narrow caroncomb.case caroncomb.narrow tildecomb.case tildecomb.narrow doublegravecomb.case doubleacutecomb.case brevecomb.case brevecomb.narrow invertedbrevecomb.case invertedbrevecomb.narrow ringcomb.case hookcomb.case dieresiscomb.case dieresiscomb.narrow dotaccentcomb.case macroncomb.case macroncomb.narrow tonoscomb.case dialytikatonoscomb.case tildebelowcomb.narrow dotbelowcomb.cap commaaccentcomb.cap
Complete Composites
Decompose
Bearings LS: -700
Width (fixed, 0) Both Sides
Left Side Bearing Point at x=0

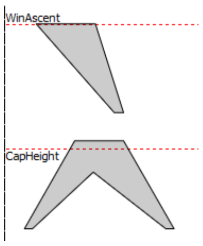
Stacking Diacritics

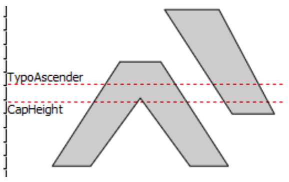


To create a font with full support for the Latin Extended Additional character set, which is essential for Vietnamese users, one needs to include stacking diacritics. This script will insert unmapped glyphs, and compose them from uppercase diacritics (or from combining or modifier diacritics if the former do not exist). They should be designed to fit below WinAscent and above the uppercase vowels. Regular diacritics will need editing to reduce their vertical height, while keeping their weight the same.

Script: Unmapped Stacking Diacritics.xml
Insert Glyphs dieresiscomb_macroncomb.case macroncomb_dieresiscomb.case dotaccentcomb_macroncomb.case dieresiscomb_acutecomb.case dieresiscomb_gravecomb.case macroncomb_acutecomb.case ringcomb_acutecomb.case ringcomb_acutecomb macroncomb_gravecomb.case tildecomb_macroncomb.case dieresiscomb_caroncomb.case tildecomb_dieresiscomb.case caroncomb_dotaccentcomb.case tildecomb_acutecomb.case circumflexcomb_acutecomb.case circumflexcomb_gravecomb.case circumflexcomb_tildecomb.case brevecomb_tildecomb.case brevecomb_acutecomb.case brevecomb_gravecomb.case brevecomb_hookcomb.case circumflexcomb_hookcomb.case
Complete Composites
Decompose
Bearings LS: -1000
Width (fixed, 0) Both Sides
Left Side Bearing Point at x=0







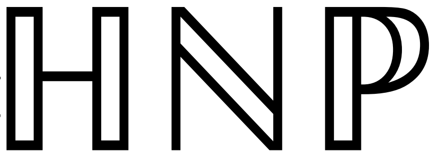
After running this script, the advance width for stacking diacritics will be set to zero. This helps in the design process as one can display an uppercase vowel in the Comparison Toolbar to see how they will look when used in composites. Complete Composites will centre them horizontally above each vowel, without moving them vertically. Some accents may need horizontal adjustment. The circumflex with grave accent in Gentium (above right) is designed with the grave on the left, in Times New Roman (above left) the accents are stacked vertically. CompositeData.xml follows the advice of a Vietnamese user with the grave accent on the right. Most fonts follow the design of Gentium.

If the vertical space between CapHeight and WinAscent is insufficient, WinAscent must be raised before designing these accents. If any glyphs exceed the WinAscent (or WinDescent), they will be clipped in most applications. Go to Font Properties, Metrics, and select Maximum, Calculate to fix it.

Spaces

This script inserts all of the fixed width spaces in the General Punctuation character set. Open Office and some other applications allow the user to use these special spaces for typesetting. After running the script there is nothing to be done, unless one wants to change the recommended advance width for thin space (1/5 em) or hair space (1/10 em). Punctuation space and figure space use the advance width for the period (46) and figure zero (48) respectively.

Letter-like Symbols



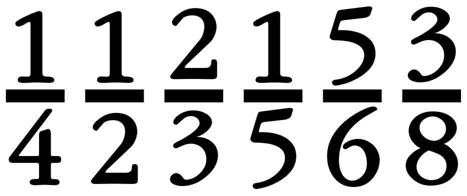
This transform script will add several letter-like symbols. It is just the initial step in the design process. Most of the glyphs will need editing to adjust weight and spacing. Some glyphs need to be rotated. Double-struck capitals should look like these glyphs from Lucida Sans Unicode, not like the simple outline letters produced by the hollow transform. Cut the outer contour to the clipboard to work on the inner contour, before pasting the outer contour back.

Script: Letterlike Symbols.xml
Insert Characters 8448, 8449, 8451-8454, 8457, 8462, 8468, 8470, 8471, 8478, 8480, 8481, 8482, 8486, 8487, 8490, 8491, 8498, 8505, 8506, 8507
CompleteComposites
Decompose
Insert Characters 8450, 8461, 8469, 8473, 8474, 8477, 8484 (Apply subsequent features)
CompleteComposites
Decompose
Hollow (50, 50) preserve bearings
Left Side Bearing Point at x=0

Stacking Fractions

Stacking fractions, alternative fractions, or nut fractions are designed to save space when typesetting fractional measurements. Regular fractions like ¼ and ½ use the fraction slash separator, while stacking fractions use a horizontal divisor.

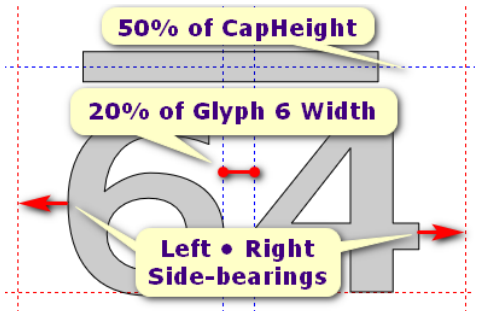
Script: Unmapped Alternative Fractions.xml
Insert Glyphs divisor.afrc onenumerator.afrc twodenominator.afrc threenumerator.afrc fourdenominator.afrc fivenumerator.afrc sixdenominator.afrc sevennumerator.afrc eightdenominator.afrc ninenumerator.afrc
Complete Composites
Decompose
Bold (7, 5)
Move (0, 5)
Center Glyph
Set Left Side Bearing Point at x=0



Insert Glyphs sixteendenominator.afrc thirtytwodenominator.afrc sixtyfourdenominator.afrc elevennumerator.afrc thirteennumerator.afrc fifteennumerator.afrc seventeennumerator.afrc nineteennumerator.afrc twentyonenumerator.afrc twentythreenumerator.afrc twentyfivenumerator.afrc twentysevennumerator.afrc twentyninenumerator.afrc thirtyonenumerator.afrc thirtythreenumerator.afrc thirtyfivenumerator.afrc thirtysevennumerator.afrc thirtyninenumerator.afrc fortyonenumerator.afrc fortythreenumerator.afrc fortyfivenumerator.afrc fortysevennumerator.afrc fortyninenumerator.afrc fiftyonenumerator.afrc fiftythreenumerator.afrc fiftyfivenumerator.afrc fiftysevennumerator.afrc fiftyninenumerator.afrc sixtyonenumerator.afrc sixtythreenumerator.afrc
Complete Composites

Insert Glyphs onequarter.afrc onehalf.afrc threequarters.afrc onethird.afrc twothirds.afrc onefifth.afrc twofifths.afrc-fourfifths.afrc onesixth.afrc fivesixths.afrc oneseventh.afrc twosevenths.afrc-sixsevenths.afrc oneeighth.afrc threeeighths.afrc fiveeighths.afrc seveneighths.afrc oneninth.afrc twoninths.afrc fourninths.afrc fiveninths.afrc sevenninths.afrc eightninths.afrc tendenominator.afrc onetenth.afrc threetenths.afrc seventenths.afrc ninetenths.afrc onesixteenth.afrc threesixteenths.afrc fivesixteenths.afrc sevensixteenths.afrc ninesixteenths.afrc elevensixteenths.afrc thirteensixteenths.afrc fifteensixteenths.afrc onethirtysecond.afrc threethirtyseconds.afrc fivethirtyseconds.afrc seventhirtyseconds.afrc ninethirtyseconds.afrc eleventhirtyseconds.afrc thirteenthirtyseconds.afrc fifteenthirtyseconds.afrc seventeenthirtyseconds.afrc nineteenthirtyseconds.afrc twentyonethirtyseconds.afrc

twentythreethirtyseconds.afrc twentyfivethirtyseconds.afrc twentyseventhirtyseconds.afrc twentyninethirtyseconds.afrc thirtyonethirtyseconds.afrc onesixtyfourth.afrc threesixtyfourths.afrc fivesixtyfourths.afrc sevensixtyfourths.afrc ninesixtyfourths.afrc elevensixtyfourths.afrc thirteensixtyfourths.afrc fifteensixtyfourths.afrc seventeensixtyfourths.afrc nineteensixtyfourths.afrc twentyonesixtyfourths.afrc twentythreesixtyfourths.afrc twentyfivesixtyfourths.afrc twentysevensixtyfourths.afrc twentyninesixtyfourths.afrc thirtyonesixtyfourths.afrc thirtythreesixtyfourths.afrc thirtyfivesixtyfourths.afrc thirtysevensixtyfourths.afrc thirtyninesixtyfourths.afrc fortyonesixtyfourths.afrc fortythreesixtyfourths.afrc fortyfivesixtyfourths.afrc fortysevensixtyfourths.afrc fortyninesixtyfourths.afrc fiftyonesixtyfourths.afrc fiftythreesixtyfourths.afrc fiftyfivesixtyfourths.afrc fiftysevensixtyfourths.afrc fiftyninesixtyfourths.afrc sixtyonesixtyfourths.afrc sixtythreesixtyfourths.afrc
Complete Composites



This script inserts unmapped glyphs, scaling the nominators and denominators from figures. It applies a bold transform and centres the glyphs. The advance width for the divisor, and thus for all single figure numerators and denominators, adds side-bearings of 100 funits to either side of the underline glyph after scaling it down by 50% to create the divisor. It is centred vertically at 50% of the CapHeight.

Nominators and denominators from 11 to 63 all derive their width from the 1/64 denominator, the side-bearings are those of the six and four denominators. The two glyphs are spaced by 20% of the glyph width of the 6 denominator. Edit the 1/64 denominator, then recompose the others if you wish.

Ordinals

The Ordinals Feature in OpenType fonts uses superscript letters automatically after numerals e.g. 1st, 2nd, 3rd, 4th, 1^{èm} 2^{o} 3^{ú} etc. This script inserts unmapped superscripts for lowercase and uppercase.

Insert Glyphs A.sups-Z.sups a.sups-z.sups
Complete Composites
Decompose
Scale (70, 68) Fixed point (0,0)
Bold (25, 8)
Move (0, 508)
Left side-bearing point at x=0
Insert Glyphs gravecomb.sups acutecomb.sups Egrave.sups Uacute.sups egrave.sups uacute.sups
Complete Composites

This script inserts glyphs for A-Z, a-z , grave, acute. It scales them by 70% about the origin and makes the glyphs bolder to compensate. It then moves the glyphs up so that the tops of ascenders in the superscripts align with the tops of ascenders. This value will need adjusting to suit each font. Complete Composites positions the diacritics horizontally and vertically to suit the size of the superscripts.

Select Glyphs for Tagging

The override range features can be used to select specific glyphs to tag them or to perform glyph transformations. Glyphs can be selected using Override Range by Code-point(s), or Override Range by Glyph Name(s).

Override Range by Glyph Names numbersign dollar plus less equal greater cent sterling currency yen logicalnot plusminus paragraph multiply divide
Width (Fixed 1303) Both Sides
Center Glyph
Left Side Bearing Point at x=0

The above script selects some maths and currency symbols, applies a fixed width, centres the glyphs, and sets the left side-bearing at zero. Add or remove glyph names to the range and adjust the width to suit the figure width of your font before running it.

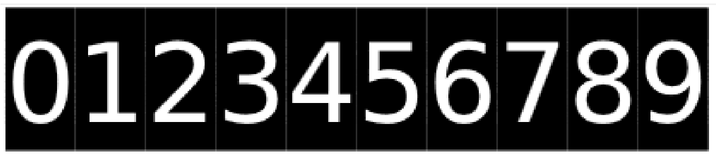
The script shipped with FontCreator selects over a thousand glyphs to tag to exclude from export to Web Fonts. Edit the glyph list and save the script with a new name to suit your needs. Glyph names can be added easily by copying selected glyphs in the glyph overview, then pasting them into the Transform Wizard dialogue. The validate button will replace line-breaks with spaces, and count the glyphs.

Click the Validate button to count the number of glyphs and check for errors

Inverse

This simple script has no parameters. It adds a black rectangle to each selected glyph, and reverses the direction of the contours to produce a white glyph on a black background.

```
Override Range by Glyph Name(s) zero-nine
Inverse
```



This script would select the figures, add a black rectangle the width of the glyph and the full height between WinDescent and WinAscent, reversing any contours to create a white glyph on a black background.

Join Contours

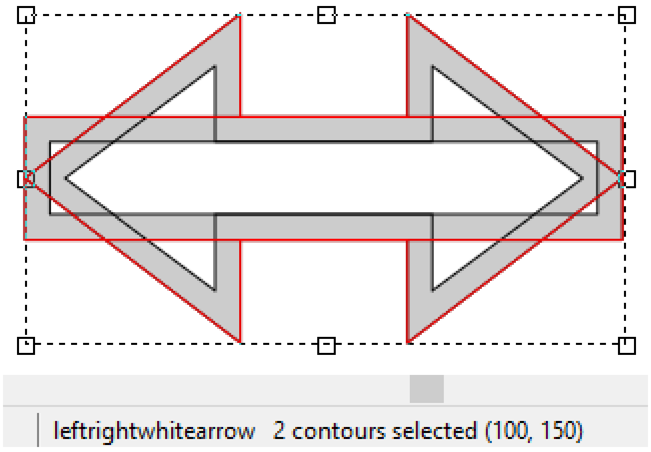
This script demonstrates how to join contours after decomposing composite glyphs. Often, one needs to adjust composite glyph members before joining them, but if Complete Composites is known to position them correctly one can join them at once using this command.

```
Insert Glyphs squareimageof squareoriginalof squareimageoforequalto squareoriginaloforequalto
squarecap squarecup righttack lefttack downtack assertion models true forces
tripleverticalbarrightturnstile doubleverticalbardoublerightturnstile notsquareimageoforequalto
notsquareoriginaloforequalto squareimageofornotequalto squareoriginalofornotequalto
elementofwithlonghorizontalstroke elementofwithverticalbaratendofhorizontalstroke
smallelementofwithverticalbaratendofhorizontalstroke elementofwithdotaccent
elementofwithoverbar smallelementofwithoverbar elementofwithunderbar
elementofwithtwohorizontalstrokes containswithlonghorizontalstroke
containswithverticalbaratendofhorizontalstroke smallcontainswithverticalbaratendofhorizontalstroke
containswithoverbar smallcontainswithoverbar znotationbagmembership
Complete Composites
Decompose Composites
Join Contours
```

This script inserts some glyphs for Mathematical Operators, composes them from other glyphs such as minus and vertical bar, then joins them with Get Union of Contours.

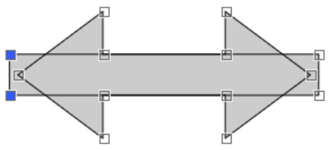
Rotate

Rotates glyphs to any angle — negative values rotate clockwise. The centre of rotation can be the centre of bearings, the corner or middle of the glyph, or any fixed point. The following script inserts several white arrows, rotates them 45° clockwise or anticlockwise, then composes black arrows based on white arrows. Comments describe what to do after running the script.

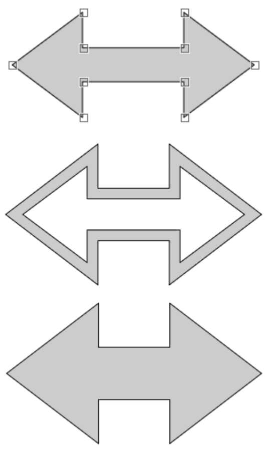


leftrightwhitearrow 2 contours selected (100, 150)

```
Insert Glyphs northeastwhitearrow
Complete Composites
Decompose Composites
Rotate -45.00 deg. M
Insert Glyphs northwestwhitearrow
Complete Composites
Decompose Composites
Rotate 45.00 deg. M
Insert Glyphs southeastwhitearrow
Complete Composites
Decompose Composites
Rotate -45.00 deg. M
Insert Glyphs southwestwhitearrow
Complete Composites
Decompose Composites
Rotate -45.00 deg. M
Insert Glyphs leftrightwhitearrow
Complete Composites
Decompose Composites
Comments
1) Cut outer contours to the clipboard
2) Move the end nodes and get union of contours with the inner contours
3) Move inner contour out of the way e.g. (Shift+Up cursor ten times)
4) Paste back the outer contours
```



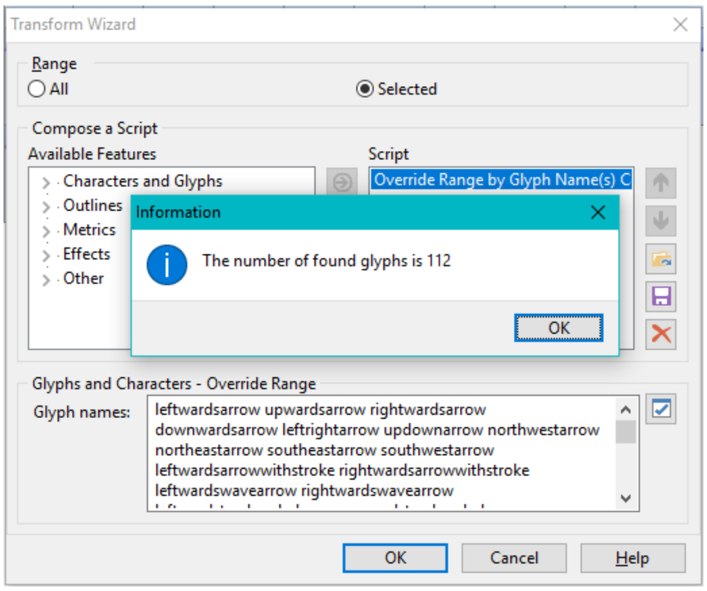
5) Move the end nodes and get union of contours with the outer contours
6) Move the inner contour back to where it was before.
There should now be one black and one white contour that do not overlap.
Insert Glyphs leftwardsblackarrow upwardsblackarrow downwardsblackarrow northeastblackarrow northwestblackarrow southeastblackarrow southwestblackarrow leftrightblackarrow updownblackarrow
Complete Composites
Decompose Composites
Comments
Recompose the Left Right Black Arrow after editing the Left Right White Arrow
Delete the inner (white) contours to make the arrows black.



Other Features

Insert Glyphs

Insert glyphs based on glyph names without mapping them to any code-point. Copy selected glyphs in the glyph overview and paste the glyph names directly into the Transform Wizard dialogue. The screen shot shows all 112 arrow glyphs pasted and then validated. Generate Glyph Names first to get user-friendly glyph names.



Transform Wizard

Range
○ All [✓ ● Selected]

Compose a Script
Available Features
> Characters and Glyphs
> Outlines
> Metrics
> Effects
> Other

Script
Override Range by Glyph Name(s) C

Information
The number of found glyphs is 112
OK

Glyphs and Characters - Override Range
Glyph names: leftwardsarrow upwardsarrow rightwardsarrow downwardsarrow leftrightarrow updownarrow northwestarrow northeastarrow southeastarrow southwestarrow leftwardsarrowwithstroke rightwardsarrowwithstroke leftwardswavearrow rightwardswavearrow

OK Cancel Help

Mirror

Mirror the glyph horizontally or vertically, optionally preserving the current side bearings.

Optimize

This script has no parameters. It reduces the number of points of all contours in the selected glyphs. Running this script may have a subtle effect on curves so check that the results are satisfactory before saving the changes. This command can be used to remove off-curve extremes introduced by other commands such as skew or rotate. Use the Background Image toolbar with Fill Outlines disabled to easily compare the results before and after this transformation. Optimize is also available from the right-click context menu in the glyph edit window when in contours mode.

Position

Use this command to move contours to a specified position. Set the position of any corner or middle of the glyph, move to a horizontal or vertical position, or do both. Glyphs must be decomposed first.

Skew

Skew the contours horizontally or vertically about the centre of bearings, middle or corner of the glyph, or about a fixed point. Can be used to create oblique glyphs.

Editing and Saving Scripts

Transform scripts are plain text XML files. Edit them in the Transform Wizard dialogue, or use a text editor like Notepad2. Be careful, though, as scripts won't run at all if the syntax is wrong. Click on the Open folder icon to load existing scripts, and click on the Save icon to save customised or new scripts. Click on the delete icon to clear the current script (it will not delete anything on the hard drive). To replace a default script with an edited version, use the save icon and overwrite the existing script.

When inserting characters or glyphs, the Validate con will check that code-points are valid, and count the number of characters or glyphs that will be inserted. There is a limit of 1,000 characters to the insert characters command, but this feature can be used several times in one transform script. Inserting glyphs is unlimited. A few additional features can be found on the left of the Glyph Transform Wizard dialogue. Most have several options.

About the Private Use Area

There are 6,400 code-points in the Private Use Area BMP (Basic Multilingual Plane) from 57344 (E000) to 63743 (F8FF). As the name suggests, font designers can use this area however they wish. They are intentionally left undefined by the Unicode Consortium so that third parties can define their own characters without conflicting with any Unicode assignments.

FontCreator 12 no longer uses Unicode mappings for its Complete Composites feature with the exception of a few Nordic glyphs proposed by the Medieval Unicode Font Initiative. If you need any more characters without assigned code-points, add them to the PUA.

Script: Nordic.xml
Insert Character(s) 57497, 57868, 58521, 58892
Complete Composites

Using Unmapped Glyphs

Glyphs that are only used for font construction, by composites, or by OpenType features, do not need to be mapped. It is recommended not to map them. FontCreator 12 ships with Transform scripts that insert Unmapped glyphs using glyph names. The Complete Composite feature uses unmapped glyphs, relying on the feature to Generate User-friendly glyph names.

If the names are incorrect, the features will not work as expected, if at all.

The Transform scripts ensure that the correct name is used. If there is any typo in the glyph names Complete Composites won't be able to find it.

Unmapped glyphs have been used to avoid possible conflicts with fonts having other glyphs assigned to code-points in the Private Use Area. Note, however, that the unmapped glyphs will not usually be available to those applications that do not support OpenType features.

The Transform scripts shipped with FontCreator are saved in:

C:\Program Files\High-Logic FontCreator\Transform

CompositeData.xml and anchorbased.xml are saved in:

C:\Program Files\High-Logic FontCreator\Composites

In the FontCreator Options dialogue, on the Advanced Tab, click on **Copy Data Files to User Data Folder,** then **Open User Data Folder** to locate the copies. Edit these copies to suit your needs; leaving the originals in Program Files untouched.

CompositeData.xml is a plain text file that can be edited in any suitable text editor to customise the Complete Composites feature in any way that users wish. I use Jarte — a Wordpad clone –- as it has all of the features that I need while being more like a word-processor than a code-editor.

Please see the Complete Composites Tutorial for some suggestions.