user manual

version 3.2

CR8 Software Solutions Limited

cr8software.net

Contents

Introduction

The art of Typeface design has been around for centuries, and many of the fonts in use today were first designed hundreds of years ago (Garamond typeface, for example was created in the 1500's by a gentleman by the name of Claude Garamond). Back then fonts were cut into steel - today you have the convenience and ease of modern computer software.

Type light 3.2 is a basic, freeware font editor, that has been designed to make it easy for a beginner to get started in the process of making and editing fonts.

Type 3.2 full version includes a full range of drawing tools, integrates an autotracing feature, vector image import, and advanced options: kerning, automated action scripts, hinting and OpenType features. Download the trail version here: **http://cr8software.net/type.html**.

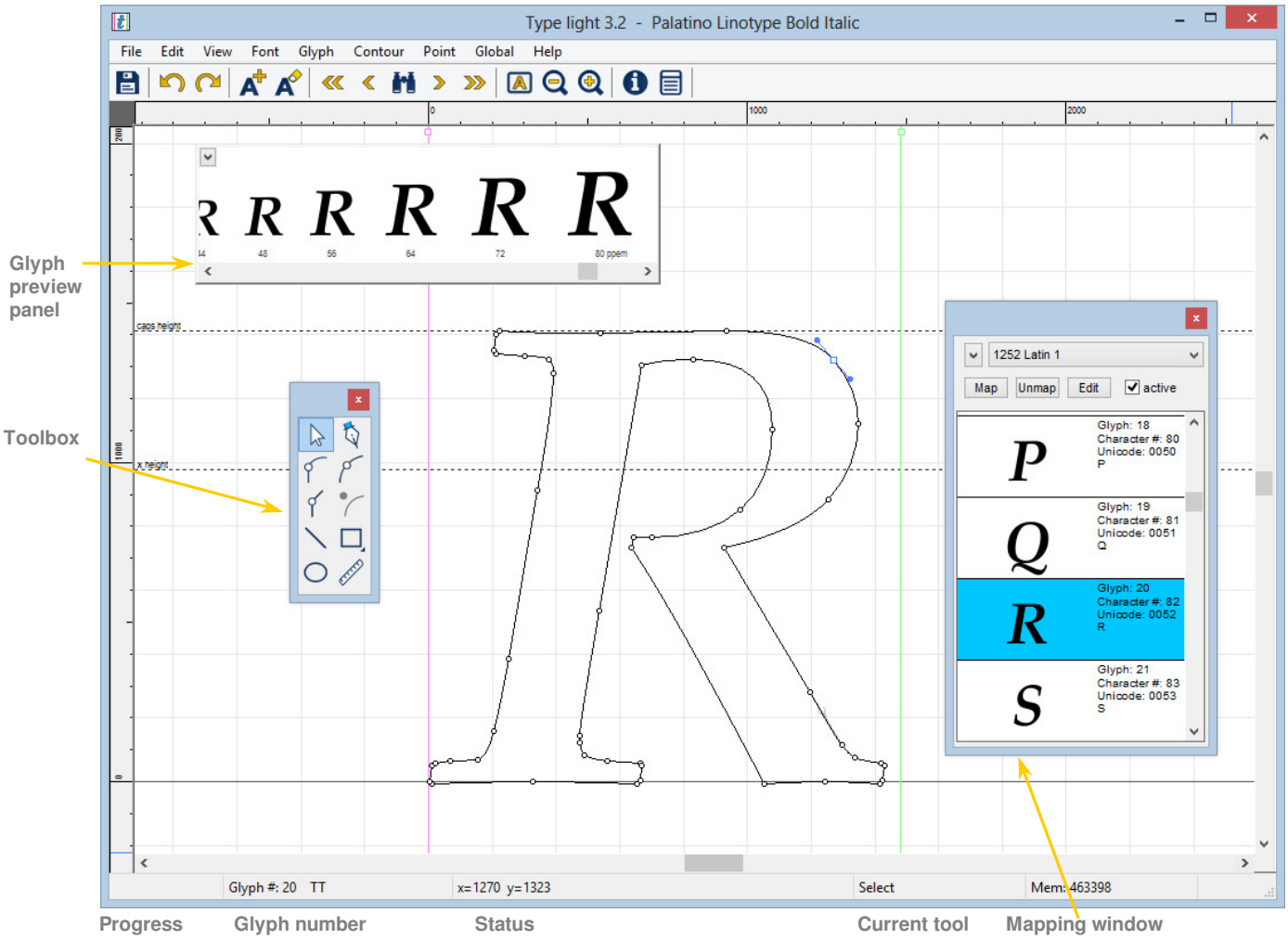
This manual is intended to be an instruction manual for Type light 3.2 rather than a tutorial or introduction to font design. If you are new to making fonts then it is recommended that you read Appendix II & III of this manual first, and also check out the links below.

For more information:

Visit **http://cr8software.net/truetype.html** for tutorials, articles and links about making fonts (general as well as specialised Type 3.2 articles).

There is a public discussion forum here: **http://cr8.proboards.com** where you can post questions or suggestions, or just to see if someone has already asked your question.

For free support post your private query here: **http://cr8software.net/support.html** or send an email to: **support@cr8software.net**.



Navigate around the edit window (main display area), by using the **scroll bars** or by **right clicking** on a clear area to grab and drag the window.

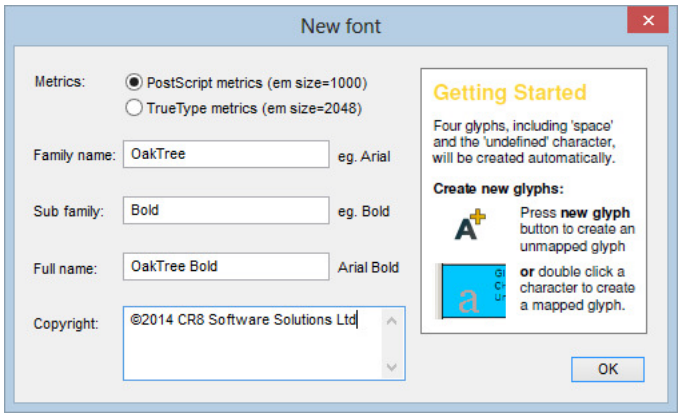
Zoom in and out, using the **magnify** and **reduce buttons**, the **+ key** and **- key**, or the mouse scroll wheel.

The magenta (p1) and green (p2) vertical lines mark the width of the glyph (see Appendix II). Move these by dragging at the top of each line, or select **glyph metrics** from the **glyph menu**.

The dashed horizontal lines are: **ascent**, **descent**, **caps-height** and **x-height**. Anything that is not between the ascent and descent line will be 'clipped'. The values of these parameters can be set using **metrics** from the **font menu** (WinAscent, WinDescent, Caps height and x-height). Do not confuse ascent and descent with typographic ascender and descender mentioned in Appendix II).

Creating a New Font

Select **new** from the **file menu**. You will be prompted for the names of your new font. All fields are required. These are:



Family name: Arial for example

Sub family: Regular, Bold, Italic etc.

Full name: Family name + Sub family (eg Arial Bold). For a Regular font, just put the Family name (Arial)

Copyright: Your copyright details.

The first four glyphs will be automatically created and mapped (see Appendix II for more details). The first glyph always represents the **undefined** character – normally an empty box-like symbol. You can edit the symbol, but you cannot map this glyph to anything (mapping to the undefined character is equivalent to unmapping).

The next two glyphs are mapped to certain **control characters,** and the last is mapped to the **space** character. This glyph must be left blank, but you can change its width to match the spacing of your font.

Press the **new glyph button** to start creating a new font. 

or

Double click on one of the characters in the **mapping window** and say **yes** to the prompt – a new glyph mapped to that character will be created.

You may now proceed to create glyphs using the range of tools available in the **toolbox.**

Common Issues

The following are a list of tips to help you avoid some common issues that can occur when creating or editing fonts.

- Do not open font files for editing from the Windows font folder. Copy them to another folder before opening them.
- Don't save font files directly into the Windows fonts folder (they will not be installed correctly).
- The correct way to install fonts is to drag the font file (save it somewhere else first) into the Windows font folder.
- If you are editing an installed font, then you will need to uninstall the original or give the new font a different font **family name** before you can install the new font.
- Make frequent backups (always good practice) of your font files, preferably as .gfs files, during the font development process.
- When a font does not work as expected, it is often a **naming** (conflicting font names) or **encoding** (make sure codepage Latin 1252 is active) issue.
- You should only have up to four different fonts that have the same font **family name** –regular, bold, italic and bold italic.
- If you change the name of a font, make sure that you change the advanced names also. Some programs will use advanced names (eg unique font name) to distinguish fonts, so they need to be unique.
- If you change the name of a font, make sure that you change the names for Macintosh (Roman) (otherwise the font may not work if installed on an Apple Mac) and for other active languages.
- Avoid creating glyphs with overlapping contours. The glyph will display correctly on a TrueType (.ttf) font (but is not recommended) but will have a white space in the overlapping region on a PostScript (.otf) font.
- When creating glyphs where one contour is enclosed by another (eg. the inner and outer contours of an 'o'), the inner contour must be in the opposite direction (clockwise or anticlockwise) to the outer contour (see appendix II).

Definitions

Glyph – Glyphs are the shapes and symbols that you design. They normally represent characters or components of characters. A font contains a list of glyphs indexed by a glyph ID. They can be in an arbitrary order, but creating them in character set order is normal practice, and has some advantages.

Character – Characters are the basic symbols that are used to represent a language. The letter A is a character in the Latin Alphabet for example.

Mapping – Mapping is the process of connecting glyphs to characters. Glyph ID #36 may be assigned to represent the letter A character in a certain font for example. A single glyph can be mapped to more than one character, and (using OpenType features) several glyphs can also be used to represent the same character.

Unicode characters – Unicode is an international character encoding system that assigns a code to every character for most of the worlds language systems. A **Unicode script** is a range containing related characters. For example the Basic Latin script is the unicode range 0000 to 007F.

Character set – A Character set is the group of characters used to represent a particular language. Single byte character sets (**Code pages**) can contain up to 256 characters. The characters can represent different languages depending on the encoding used. Fonts can contain more than one Code page, which can be selected from the font selection menu on most programs. (Western = Latin 1252) .

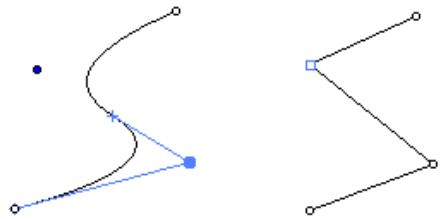
TrueType Points View

Select **TT points** from the **view menu**. This is the native TrueType format and can only be used with TrueType curves (see Appendix III). A TrueType font stores glyphs as a series of contours made of points – either off-curve (dark blue dot/light blue dot when selected) or on-curve (white circle/light blue square when selected).

The gray arrow between point one and point two on each contour indicates the contour direction (see Appendix II regarding correct contour direction).

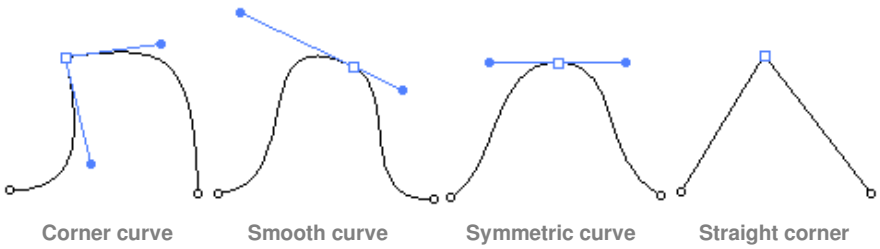
Point one must always be an on-curve point.

Using on-curve and off-curve points is the correct representation of a TrueType glyph, but it is not necessarily the easiest way to design glyphs.



Nodes View

Select **nodes** from the **view menu**. Another way to represent a glyph is by using nodes (white circle/light blue square when selected) which are always on-curve, and control points (light blue dot) which are always off-curve. The light blue line between the control point and the node is at tangent to the curve. Move the position of the control points to change the shape of the curve.*



Corner curves have control points that can be moved independantly (unlinked). Smooth and symmetric curves have linked contol points that are always in line with the node. Straight lines do not have any control points.

Use the point menu (or right click) to change the type of curve.

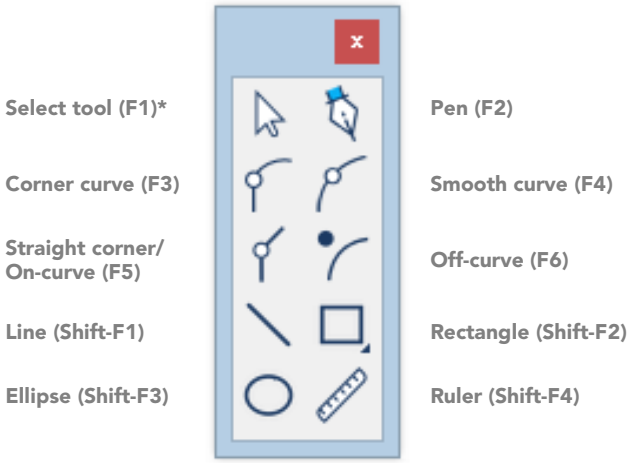
* Other font editors only allow this type of editing with PostScript curves – Type light 3.2 also allows you to edit TrueType curves in this manner, but limits are imposed to restrict the outline to a TrueType curve. ***Because of these limits, the controls may not behave as you may expect when editing TrueType curves in this mode - nodes other than the one you are moving, may also move, for example.***

The **toolbox** contains the tools that you will need for creating and drawing glyphs.

Using different tools, you can move points and contours, draw and manipulate lines and curves, create shapes, and measure distances.

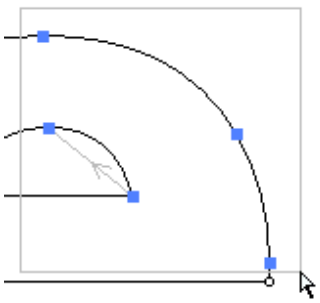
Some tools have extended functions or options, indicated by a small triangle in the lower right corner. Right click to access these options.

*The keyboard shortcut keys for the Toolbox are the function keys F1-F6.



The Select Tool

You can use the **select tool** to highlight and move points around.



Drag a rectangle over points or nodes to select multiple points. Pressing **shift** at the same time allows you to select more points, either by drawing another rectangle, or by clicking on individual points. Pressing **ctrl** at the same time as selecting points will select the entire contour.

Multiple point selection allows you to move many points (move one of the selected squares to move all of them, or use the cursor keys), perform transformations (see below) and allows other contour operations using the **contour menu**.

Selecting a point by clicking on it allows you to move a single point, and gives you access to the options on the **points menu** – either from the main menu or by clicking the right mouse button.

If you are in **nodes view mode** then when you select a node, the control points belonging to that node will be displayed. The control points (light blue dots) control the curve - click and drag them to alter the shape of the curve (press **Ctrl** to restrict movement horizontally and vertically). Control points can be removed by dragging them into the node, and then releasing the mouse button (PostScript curves only). Clicking on a node while pressing **shift** allows a new control point to be dragged out of the node (PostScript curves only).

When editing a glyph with TrueType curves, moving a conrol point may also affect neighboring nodes. PostScript curves do not have this limitation.

The Pen Tool

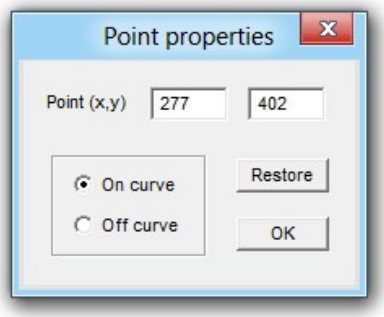
The **pen tool** will draw smooth curves. Left mouse click to add a node, and drag the mouse before you release the button to extend the control points. The curves will be symmetric about the node. If the **shift** key is pressed then the control points will no longer be linked, and you will be able to form smooth or corner curves.

If you are editing a glyph with TrueType curves, then when you release the button the curve will convert to a TrueType curve (with additional nodes).

The Node and Points Tools

These four tools work as follows:

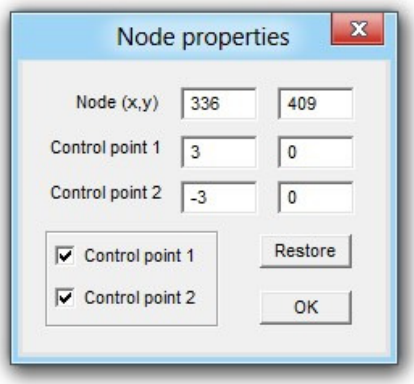
- If no point or node is currently selected, clicking will start a new contour.
- If the last point or node on a contour is selected, then clicking will add a node or point to the current contour.
- Clicking on the first point or node of a contour will close the contour.
- If the mouse pointer is over a curve or line, then a small box will appear next to the mouse pointer indicating that clicking will insert a new point or node in the middle of the contour.



The **corner tool** and **curve tool** allow you to adjust the degree of curve. If you move the mouse before you release the button, you can change the position of the control points.

The **off-curve tool** functions only when editing a glyph with TrueType curves.

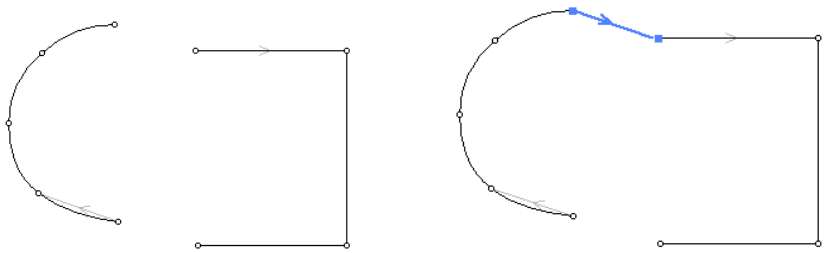
Points and nodes can be precisely positioned by using **properties** from the **points menu** (or right click, **proporties**).



Opening and joining contours

A **closed** contour can be **opened** by selecting the first node and then deleting it (you are actually deleting the last node, which is on top of the first node for a closed contour).

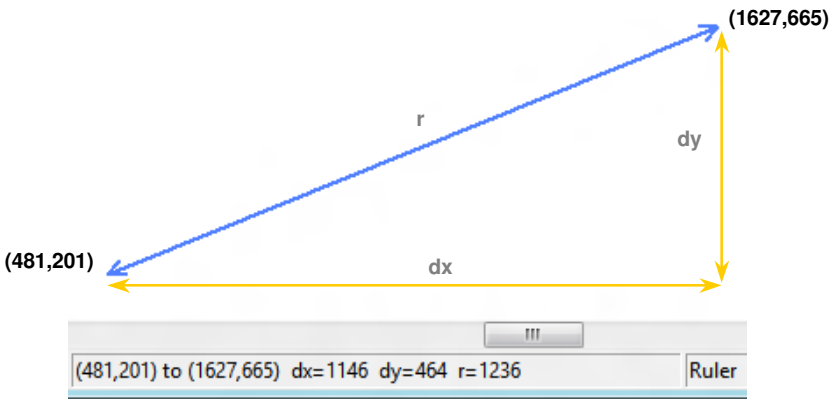
Two open contours can be joined together by selecting the last node of one contour, and drawing to the first node of the other (the small gray arrow indicates contour direction). You can do this using the pen, corner curve, smooth curve, on-curve, off-curve or line tools.



1 2

The Ruler

The **ruler tool** can be used to measure distances. Click at the start position (or node) and drag to the end position (or node). Various measurements will appear in the status bar:

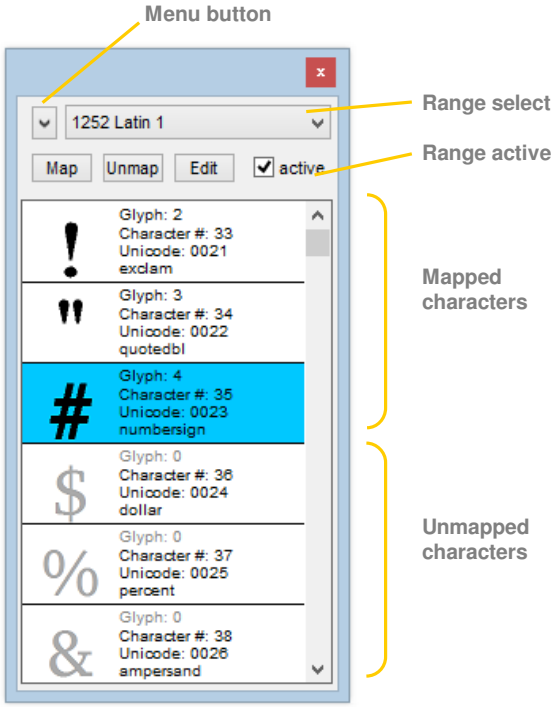


Mapping Window Modes

The **mapping window mode** toolbar button will sequence through the different modes available. You can also change the mapping mode from the **mapping menu**. The modes available are:

1 - Code Pages (character sets)
2 - Unicode Scripts (a range of unicode values)
3 - Glyph List (all mapped and unmapped glyphs)

Use **range select** to select which characters are displayed, and the **range active box** to indicate which ranges are used in your font. To make a Western character set font, for example, select the **1252 Latin 1 page range**, and map glyphs to these characters, then check the **range active box**.



Mapping Glyphs

Modes 1 and 2 allow glyphs to be mapped to unicode characters (uni0000 to uniFFFF). If the **new glyph button** was used to create the glyph, then it will need to be mapped:

To map the current glyph (the glyph in the edit window) to a character, select the character in the mapping window (it will become highlighted blue), then press the **map button**. A single glyph is normally mapped to a single character, but may be mapped to multiple characters.

Multiple glyphs can be mapped to multiple characters in one operation. Select the first character to be mapped, then, while pressing **shift** or **ctrl**, select the last character. The whole range of characters will be highlighted blue. If, for example, glyph #10 is the current glyph, then it will be mapped to the first character, glyph #11 will be mapped to the second character, etc, when the map button is pressed.

Modes 3 (glyph list) allow glyphs to be mapped to all unicode characters, including characters in the supplementary planes (uni10000 to uni10FFFF). Select a glyph or range of glyphs in the mapping window, then press the **map button**. A prompt will appear. Enter the unicode code point (in hexadecimal format) that you want to map the glyph (or first glyph in the range) to.

Mapping Menu

From the **mapping menu** you can add, select or remove **bookmarks**. A bookmark will allow you to return to your current location in a script, codepage or glyph list. To remove a bookmark, select the bookmark from the **mapping menu**, then select **remove bookmark**.

Resizing

Some glyphs (composite glyphs or glyphs of some non latin alphabets) may not fit into the default row size of the mapping window. To resize the rows:

1 - Make sure **enable row resizing** is checked in **preferences**.

2 - Select a row in the mapping window (highlighted blue).

3 - Widen the character space by clicking and dragging where the text margin is located.

4 - Increase the height of the character spacing by dragging the base of the highlighted cell.

Symbol Fonts

A **symbol font** is a special type of font normally containing pictorial type glyphs. Symbol font glyphs are mapped to unicode F020 to F0FF (corresponding to ASCII characters 20 to FF). To create a symbol font, map glyphs to the characters in the **Symbol Character Set** Code Page and set to **active**. NOTE: Setting the Symbol Character Set to active will overide any other active Code Pages, and only characters F020 - F0FF will be mapped in the output font.

An OpenType font contains a whole lot of information – names, font descriptions, parameters and metrics specific to your font. Type light 3.2 allows you to set and change most of these.

If you are new to font design, and are creating a basic font, then you can keep the default values and use the automatic settings. See Appendix II for a better understanding of various parameters.

Names

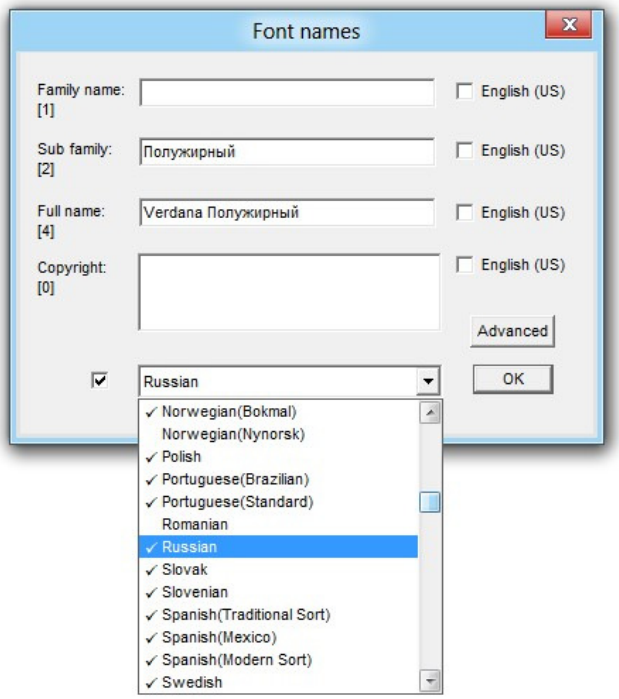
When you create a new font, you are prompted for certain name information. This information is for the default language – English (US). Your font can contain name information for other languages too. Use the **names** from the **font menu** to add more entries to the names table.

Select the language at the bottom of the window and check the box to the left to make the language active. Blank fields will not be listed in the names table . If a English (US) box is checked, then the English (US) string will be duplicated in that names table listing.

The **advanced button** allows you to enter various other names. The most important ones are: **Unique ID [3]** and **PostScript name [6]**. If you change the name of your font then you must change these also – or blank them out. If these fields are invalid or blank, then Type light 3.2 will generate valid names when you save the font.

For an ordinary Latin font, name information would normally only entered for **English (US)** and **Macintosh (Roman)** languages (these two must always be present).*

Name strings support unicode, so you can paste unicode characters into the naming window input boxes.



* Most non Latin fonts actually only have enteries for English (US) and Macintosh (Roman).

Font Metrics

Use **metrics** from the **font menu** to set your font's metrics. These parameters are certain measurements specific to your font. Some of these are self explanatory like **underline thickness** and **italic angle**. The most important ones to understand are:

EM Unit size:	Usually set to 2048 (TrueType or OpenType TT), or 1000 (OpenType PS). This is used to calculate the point size when the font is displayed.
WinAscent:	top (anything above this may be clipped)
WinDescent:	bottom (anything below this may be clipped)
Caps height:	uppercase character height
'x' height:	lowercase character height (height of a lowercase 'x')

If you have checked the option **show horizontal markers** (using **parameters** from the **view menu**) then the last four parameters above will be visible in the main edit window as horizontal lines.

Font Description

Use **description** from the **font menu** to set the parameters that describe the font: weight, width, version number, italic, bold and mono-spaced. If you set the font to **Mono-spaced** then the advance width of all glyphs (except glyph number 1 - which must have an advanced width of zero), will be constant. Advanced description parameters can only be altered using Type 3.2 full version.

A composite glyph has no contours or points of its own, but is made up of other glyphs. The composite is displayed as a blue outline, and although you can move and transform the components (as if you were manipulating contours) you cannot move individual points unless you first **decompose** the glyph by selecting **decompose** from the **glyph** menu.

Glyphs can be stored as composites in a **TrueType font (.ttf)**, but when saving an **OpenType PostScript font (.otf)** the glyphs will be not stored as composites – when you open the font again, they will be decomposed. (Save a copy as a **.gfs file** if you want to alter the glyphs as composites at a later time).

Copy and Paste Functions

Contours and whole glyphs can be copied and pasted between glyphs:

- Use **copy glyph** from the **edit menu** to copy the current glyph to the clipboard.
- Use right click **copy glyph** from the **mapping window** (glyph list mode only) to copy the selected glyph to the clipboard.
- Use **copy** from the **edit menu** to copy only the selected points to the clipboard.
- When the destination glyph is in the Glyph Window, use **paste** from the **edit menu**.
- To create a composite glyph, first make sure that **paste to create composite glyphs** is enabled in **preferences**, then paste to a blank glyph. Paste to an existing composite glyph to add another component glyph.
- The **paste metrics** option from the **edit menu** pastes only the Left Side Bearing and Advance Width from the clipboard.

When creating glyphs, it is useful to be able to see what the final glyph will look like in relation to other glyphs, and at different point sizes.

Glyph Preview Panel

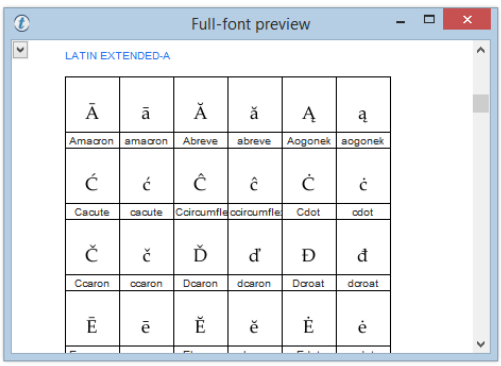
Select Glyph Preview from the View menu to display the **glyph preview panel**. Use the drop down menu button to select the preview mode:

Glyph preview –preview the current glyph at different font sizes.
Text preview –preview the current glyph with other glyphs. This is very useful when setting character spacing. Use **set preview text** to set the text displayed.

The **glyph preview panel** displays a 'live' preview – so changes made in the edit window will immediately be seen there.

Full Font Preview

Select **full font preview** from the **view menu** or press the full font preview button. This will display all glyphs in the font in a grid.



Full font preview is not 'live' – you cannot edit glyphs while viewing full font preview.

When a font is displayed at a small point size on a low resolution device (eg a screen), several distortions can occur. This happens because there are not enough pixels to accurately represent the rasterized font. Adjustments (called hints) can improve the appearance of the font at low resolutions.

Hints can either be global (apply to all glyphs within the font) or can apply to individual glyphs.

You can use Type light 3.2 to specify **gasp hinting** only. Type 3.2 full version also supports **global hinting** of **PostScript** fonts, which can be auto-hinted using Adobe's free font development kit.

Gasp (Grayscale) Hinting

Gasp hinting allows you to specify the rasterization technique for a TrueType font when it is rendered on grayscale-capable devices at different sizes.

A typical Gasp table could be:

Font size^	**Rasterization technique**
ppem<=8	grayscale only *
9<=ppem<=19	gridfit only **
20<=ppem	gridfit and grayscale

Select **gasp hinting** from the **font** menu. Enter 8 in the **max ppem** box, select grayscale in the **smoothing selection**, then press **add**. Enter 19 in the **max ppem** box, select gridfit in the **smoothing selection**, then press **add**. Enter 65535 in the **max ppem** box, select grayscale & gridfit in the **smoothing selection**, then press **add** (always enter 65535 as the max ppem for the last entry).

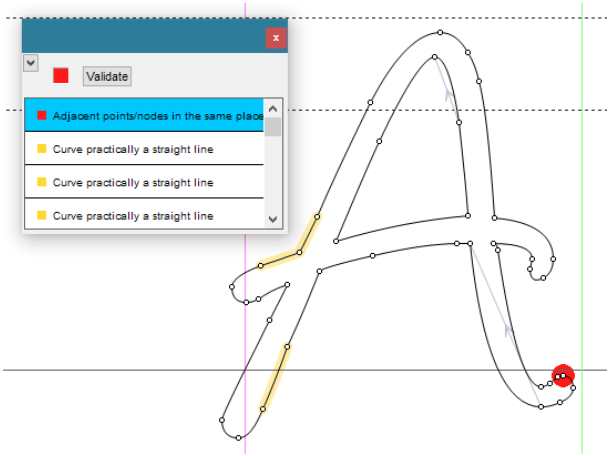
^ ppem = pixels per em. Em for a 72 point font is 1 inch (1pt = 1/72th inch).
So a 72pt font rendered on a 96 dpi (dots per inch) monitor is 96 ppem.
(So it follows that 8 ppem would be 6pt on a 96 dpi monitor).

* Note for your computer must also have font smoothing activated for grayscale rendering.
** Gridfit means use TrueType hinting instructions (they must be present in the font).

Type light's glyph validation function can detect many common glyph design errors and potential problems.

Press the glyph validation button  to validate the current glyph.
Glyphs that have been validated will be indicated on the mapping window as either: **green–ok**, **yellow-warning** or **red-error**.

Any problems are listed in the glyph validation window. Selecting a problem will highlight its location in the edit window.



The full version of Type 3.2 has an auto-repair function that will automatically repair most errors. Note that this does not exist on the light version.

Validation options can be selected from the drop down button on the validation window. This will allow selection of which errors/warnings will be checked, and which will be ignored.

- **ERRORS:** (can potentially cause the font to function or display incorrectly) Advice: Fix all these.

Glyph has too many nodes - More than about 1,500 nodes can cause problems. Typically this only occurs with auto-trace. Retrace with different settings or use **simplify curves** touch-up tool.

Self-intersecting contour – Contour crosses itself.

Open contour - Contours must be closed.

Intersecting contours – Two contours cross each other.

Duplicate contour – Same contour is duplicated on top of itself.

Contour in wrong direction* – Concentric contours should be in opposite directions.

Composite duplicate component – Same component is duplicated on top of itself.

Contour with one or two nodes/points – A straight line contour with no thickness.

Contour too small – Contour is too small to be useful, and is probably an error.

Adjacent points/nodes in the same place – Node is redundant and is on top of its neighbour.

Missing node/point at extrema – Glyphs should have a nodes/point at the four extrema (highest, lowest, left-most, right-most) points of its outer-most contour.

WARNINGS: (fixing these can possibly reduce the number of nodes used, make the glyph render better at small sizes, or may prevent other problems later on). Advice: Fix these if it is practical to do so, and if doing so does not alter the desired shape of your glyph.

Missing node at extrema (node nearby) - There is a no node at one of the extrema points, but there is a node close enough so that this should not be an issue. It may be better to leave as is, as inserting an extrema node may noticeably alter glyph shape.

Points/nodes very close together - Two points are so close together that they may be able to be merged into one without affecting glyph shape.

Curve practically a straight line - Curve is so flat that it probably should be a straight line.

Straight line almost horizontal/vertical - Straight line is almost vertical or horizontal. Making it actually horizontal or vertical may improve appearance of the glyph at small point sizes.

Almost a smooth curve - The curve is almost smooth, and probably should be.

Straight line with redundant nodes/points - A straight line contains a redundant node in the middle. This node can be removed.

Composite intersecting components - Not a problem when generating a TrueType font. If outputting as a PostScript font then this should be fixed.

Nodes/points outside clipping area - Anything above WinAscent and below WinDescent will be clipped on Windows computers.

Contour very far away - Contour is so far away that it is probably an error. Auto-repair will delete contour.

* The **validation options** dialog allows you select whether glyphs in your font have clockwise or counterclockwise outer contour direction. The default setting is auto, but it is recommended that you manually select the correct direction for your font before validation. The contour direction check cannot be performed if there are already **open contour** or **intersecting contour** errors.

Type light can be used to create colour fonts. The following colour font format is supported:

LAYER (Microsoft COLR) fonts: A colour glyph is composed of one or more other glyphs on top of each other. Each glyph represents a single colour layer.

These additional colour font formats are also supported in the full version of Type 3.2:

SVG (Adobe/Mozilla) fonts: A colour glyph consists of a single embedded SVG graphic.
APPLE (sbix) fonts: A colour glyph consists of embedded bitmap images – normally PNG.

All of the above formats allow the inclusion of a normal base glyph (or 'fallback' glyph). This glyph will be displayed if the font is rendered on a system that doesn't support the colour font format.

In Type 3.2 only the base glyph will be displayed in the **glyph preview** and **mapping windows**, but a small rainbow icon will appear next to colour-enabled glyphs in the **mapping window**.
The **full font preview window** will render glyphs in colour if the **colour on** checkbox is ticked.

Create a Colour Font

Select **colour font conversion** from the **global menu** to create a colour font. Type light will allow you to create a Colour LAYER type font only.

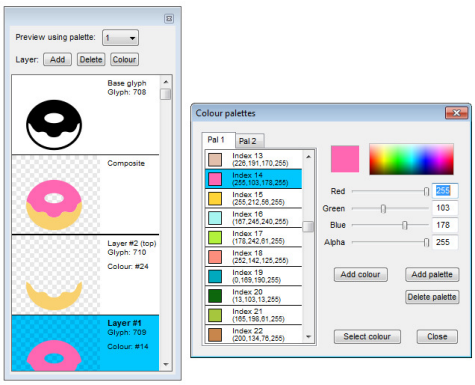
Colour Font Mode

To create/edit colour glyphs open the **colour font window** by selecting **colour font mode** from the **view menu**. This window will also be opened if you double click on a glyphs rainbow icon.

Double click on the base glyph or composite to edit the contours for the base glyph.

Double click on a layer to edit the contours for that layer.
Press **colour** to select the colour for each layer.

Use **add** and **delete** buttons to add or delete layers, use the right-click options for moving layers up or down.



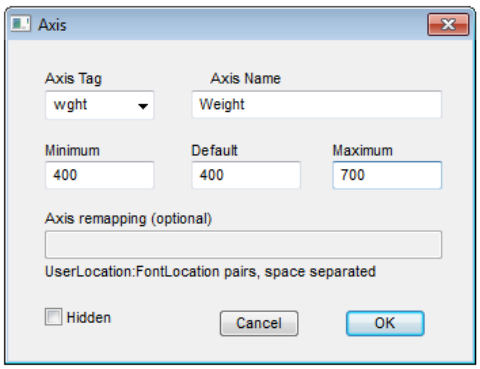
Using Colour Fonts

Support for colour fonts is growing. Adobe Photoshop and Illustrator now support colour SVG (for Mac and Windows) and colour Apple fonts (for Macs only). Many browsers also support colour font formats. For more information please visit these links:
https://www.colorfonts.wtf
https://msdn.microsoft.com/en-us/library/windows/desktop/mt790715(v=vs.85).aspx

Type 3.2 can be used to create variable fonts. Currently only TrueType variable fonts are supported. See the link[1] at the end of this chapter for an overview of variable fonts.

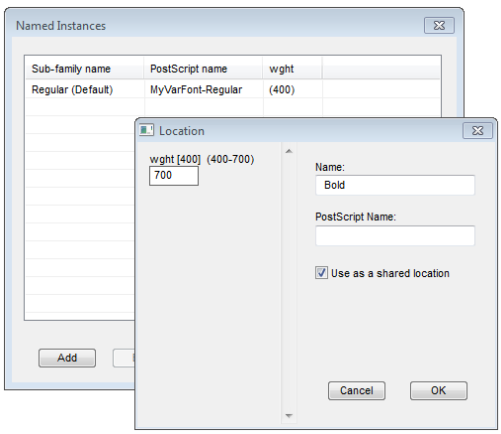
When making a variable font it is important to start with a complete (non-variable) font that represents the default instance. Editing the default instance later is more difficult.

1. Add Axes of Variation



Select **view menu/variable font mode** to open the **variable font window**. From the drop-down menu on the **variable font window** select **axis list**, then press **add**. You can select a registered axis[2] from the drop down menu, or create your own (use uppercase tags for a user-defined axis). Select suitable maximum, minimum and default values for your axis.

2. Add Named Instance(s) at Axis extreme(s)

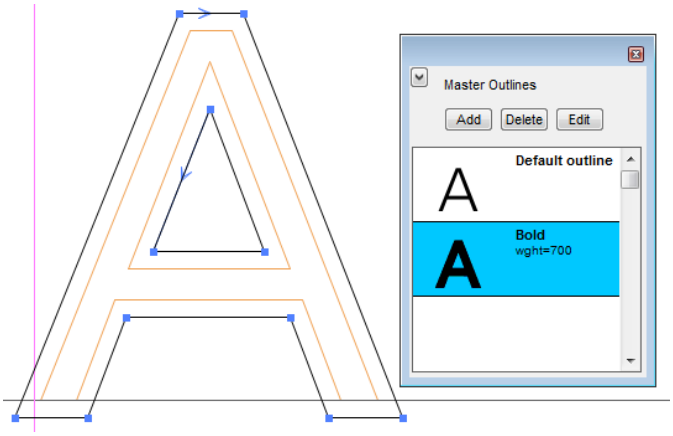


Select **named instances** from the drop-down menu and press **add**. At this stage just create named instance(s) at the extreme(s) of your axis. For example, for a weight axis running from 300 to 700 with default regular font at 400, create an **instance** at 300 named 'Light' and one at 700 named 'Bold'. You will need to create **master outlines** at these two axis locations. Check the **use as a shared location** box so that you can reference these locations later. PostScript name is optional.

3. Create Master Outlines

Select a glyph, then press **add** on the **variable font window**. Select the shared location named 'Bold' from the drop-down menu and press OK. A new **master outline** will be added to the **variable font window**. Click on this and you can edit the outline.

- The same number of contours and points must be maintained between master and default outlines on each glyph to allow interpolation. Editing tools are restricted for this reason. This button will appear on the toolbar to indicate restricted editing. Clicking will unlock some additional tools, which will allow adding/deleting points and contours of the **default glyph only** (corresponding points/contours will also be added/deleted to the master outlines).

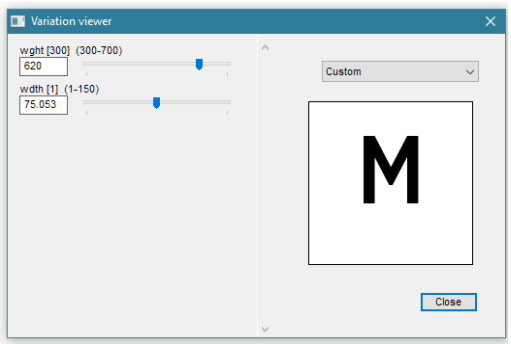


- Master outlines are stored as deltas (offsets) from the default glyph, so altering the default glyph will also alter the masters (this is why you want to have the default font complete before adding variations). The orange background outline represents the shape of the master if all of the offsets were zero (**glyph menu/clear master to default** will clear a masters offsets to zero).
- Altering a master may also alter other masters (masters that are not independent are marked with a *). Generally master outlines at the extremes of a single axis are independent, and these should be defined first.
- Overlapping contours and points (which are to be avoided in normal fonts) are allowed in variable fonts. For this reason some glyph validation checks are disabled for variable fonts.

Previewing variable fonts

Glyph preview will preview at the axis location of the current selected master outline if **show variation** is checked.

Variation viewer on the **variable font window** drop-down menu can be used to view all interpolations of a glyph using sliders for each axis or at named instances. Full front preview also has the option to view variations.



Additional Named Instances

You may want to add a **named instance** at an interpolated axis position that does not have an explicit **master outline**. In the above example, the default font is Regular (axis location=400) and we have added a Bold **master outline** at axis location 700. We can now generate any weight in-between by interpolation.

If you simply add a **named instance** called 'SemiBold' at weight axis = 600, then this will appear as a selectable option in supported applications[3] like Illustrator. Do this as per step 2 above, but don't bother making it a shared location, as you will not have to make a **master outline** there.

The SemiBold glyphs will be interpolated 2/3rds between the default (400) and Bold (600) outlines. From a design perspective, this may not be what you want. For example, maybe the outlines that are currently at 550 define a better SemiBold. In this case you can re-map the **user coordinates** of 600 to the current **font coordinates** of {550}*: open the **axis list** and enter 600:550 in the re-mapping space for the weight axis.

* Type 3.2 will place { } around an axis location to indicate a font coordinate when it is different from a user coordinate due to re-mapping.

Refences/Links

1. Overview of variable fonts (see also further references at end of article)
https://www.monotype.com/resources/articles/variable-fonts-making-the-promise-a-reality

2. Registered axes: **https://docs.microsoft.com/en-us/typography/opentype/spec/dvaraxisreg**

3. List of apps/browsers/operating systems that currently support variable fonts:
https://v-fonts.com/support

4. Useful sites for testing/viewing variable fonts:
https://v-fonts.com
https://wakamaifondue.com
https://www.axis-praxis.org

OpenType Tables

Type light 3.2 creates TrueType and OpenType TT fonts (.ttf) with these tables:

OS/2	**cmap**	**glyf**	**head**
hhea	**hmtx**	**loca**	**map**
name	**post**		

OpenType PS fonts (.otf) will have the above tables, but the **CFF** table replaces the **glyf** table.

When editing a font, the following tables can be included if present in the original file, or, in the case of the Gasp table, if it has been created by the user:

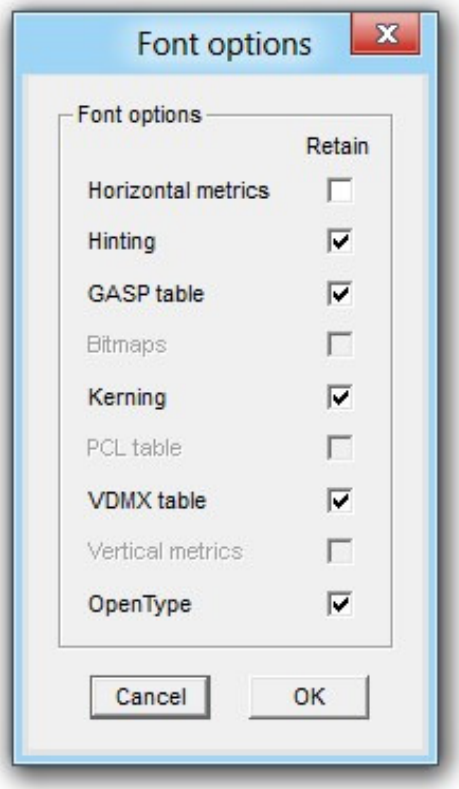
prep
cvt
LTSH
fpgm
— If 'Hinting' retained^.

EBDT
EBLC
EBSC
— If 'Bitmaps' retained.

vhea
vmtx
— If 'Vertical metrics' retained.

hdmx*
kern
gasp
PCLT**
VDMX
— Select individually.

BASE
GDEF
GPOS
GSUB
JSTF
— If 'OpenType' retained***.



^ These tables for TT fonts only. For PS fonts the hinting option will control glyph level hinting only.

* If you have edited any glyphs, then the **hdmx** (horizontal metrics) table may no longer be accurate – it is then recommended that you do not retain it. **VDMX** and **hdmx** tables can be re-created using a free tool from Microsoft called **CacheTT**. (**www.microsoft.com/typography/tools/tools.aspx)**

** If youhave changed the fonts description, then the **PCLT** table may no longer be accurate - it is then recommended that you do not retain it. The **PCLT** table is strongly discouraged for use with OpenType fonts.

*** Fonts saved as OpenType will also be given an empty **DSIG** table, This is to give the font the OpenType icon on Windows XP. The original digital signature (if any) will not be retained, as it is no longer valid for an edited font.

A Bit of History

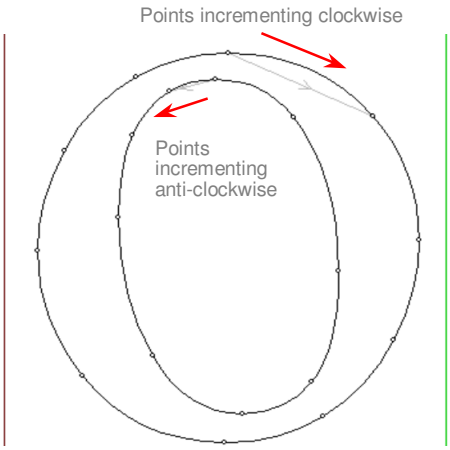
TrueType was originally developed by Apple in the late 80's, after Apple & Microsoft rejected a proposal by Adobe to use Adobe Type 1 (PostScript) fonts for their operating systems. Through a deal with Apple, TrueType was adopted by Microsoft in 1992, for their Windows 3.1 operating system.

Adobe joined forces with Microsoft in 1996 to combine their technologies and produce OpenType, which supports both TrueType and PostScript formats. Adobe finished converting its entire font library to OpenType (PostScript) fonts in 2002 with the intention that Adobe Type 1 fonts (see Appendix III) eventually be phased out.

Glyphs (TrueType)

The outlines of a TrueType glyph are defined by contours. Contours are defined by points.

Points of a contour are either on-curve (defining straight lines) or off-curve (defining a type of curve called a quadratic spline). The first point on a contour (point zero) must always be an on-curve point. See Appendix III for more details.



The filled-in area of a TrueType glyph is always on the right-hand side of the contour. So, for the letter 'o' here, the points forming the outside contour will increment clockwise, and the points forming the inner contour will increment anti-clockwise.

(Note that technically PostScript curves should be in the opposite direction - with the outside contour counterclockwise.)

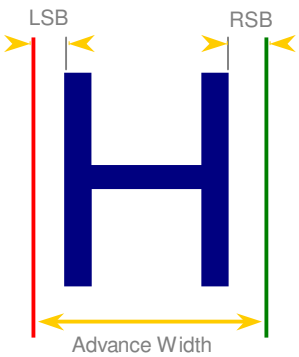
The extremes of a glyph should be defined with on-curve points.

Glyph Metrics

Certain metrics define the horizontal dimensions of a glyph. These can be set using **glyph metrics** from the **glyph menu**, or by dragging the top of the red and green vertical markers.

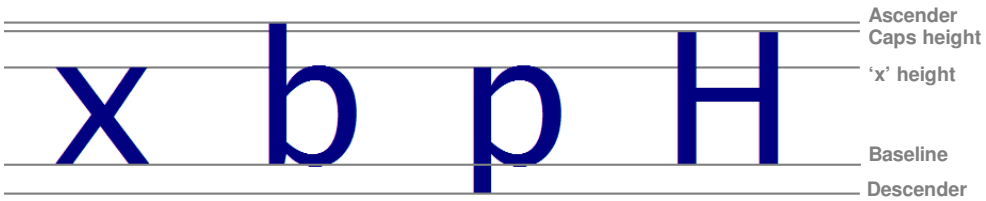
The Advance Width (AW), Left Side Bearing (LSB) and Right Side Bearing (RSB) determine the spacing between characters.

AW = LSB + character width + RSB



Font Metrics

Certain metrics define the vertical dimensions of characters in an OpenType font:



ASCENDER: Height of ascender (usually height of 'b')
DESCENDER: Height of descender (usually depth of 'p')
CAPS HEIGHT: Height of uppercase characters (defn: height of 'H')
'x' HEIGHT: Height of lowercase characters (defn: height of 'x')
BASELINE: At y=0

Do not confuse the ascender and descender with ascent and descent (called WinAscent and WinDescent in **metrics** from the **font menu**). Ascent and descent define the upper and lower limits of all glyphs – anything outside these limits will be clipped.

The **EM square** determines the size of the font when it is displayed. For example, when the font is displayed at 12 points, the EM square will be 12 points high (1 point = 1/72 inch).

The **EM square** was traditionally (from type-setting days) defined as the size of an uppercase 'M', but typically the **EM square** encompasses the ascenders and descenders with some extra (internal) leading as well.

The size of the **EM square** is usually set at 2048 units for a TrueType or OpenType TT font, and usually set at 1000 units for OpenType PS fonts.

Standard Glyphs

It is standard for a OpenType font to contain mapping to the Macintosh Roman character set, even if the font is only to be used for Microsoft Windows. Also, to meet Apple specifications, the first four glyphs of a OpenType Font should be these:

GLYPH #0 Used for **undefined** characters - normally a box shape
GLYPH #1 Special glyph with no contours, and zero width
GLYPH #2 **CR** character - no contours, but with a defined width (mapped to 0009 and 000D)
GLYPH #3 The **space** character - no contours, but with a defined width (mapped to 0020)

Visit the typography links here for more information: cr8software.net/links.html

Curves and Outlines

Outlines (the curves that form characters) are stored as a series of points, using one of two methods to mathematically describe their shape. We can say that fonts either have **PostScript outlines** or **TrueType outlines**. This appendix briefly describes the difference between the two, and the implications for font creation and editing.

Types of Outline Fonts

TrueType
extension: **.ttf**
TrueType outlines

Type 1 (not supported by Type light 3.2)
extension: **.pfb** & **.pfm**
PostScript outlines.

OpenType (TT)
extension: **.ttf**
TrueType outlines. Actually identical to TrueType fonts, but may contain additional OpenType information.

OpenType (PS)
extension: **.otf**
PostScript outlines. Essentially Type 1 fonts wrapped in a TrueType file structure.

Abode's intention is that Type 1 fonts be eventually phased out and replaced by OpenType PS fonts. Adobe Type 1 fonts are not supported by Type light 3.2.

TrueType and PostScript Outlines

Outlines in a font are described by a series of points. To describe a straight line, you only need to specify the co-ordinates of the two end points, but to specify a curve, you need some extra points in between. A mathematical equation called a bezier curve is used.

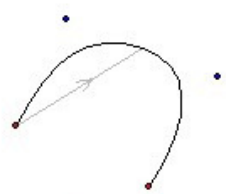


Fig 1a

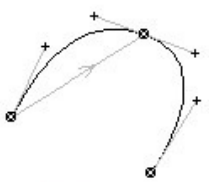


Fig 1b

Fig. 1a shows a curve described by two end points and two off-curve points. This is a quadratic bezier, or TrueType outline.

Fig. 1b shows the same curve with three nodes and their control points. This is a cubic bezier, or PostScript outline.

In fact, quadratic beziers are a subset of cubic beziers, so any TrueType curve can be converted exactly to a PostScript one (like in Figure 1). Conversion in the other direction is not so simple, and it may require several quadratic curves to approximate a particular cubic bezier.

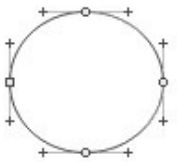


Fig 2a

Fig 2b

Fig. 2a shows a circular curve represented by a PostScript outline.

Fig. 2b, the equivalent TrueType outline, requires many more points and is only a near approximation to the original curve.

So conversion of TrueType fonts to PostScript ones is an exact science (aside from other factors such as scale and hinting), but conversion of PostScript fonts to TrueType is only a close approximation. Most commercial TrueType fonts are designed as PostScript curves, and then converted to TrueType.

Because PostScript curves are a superset of TrueType, a greater range of curves can be drawn with fewer points, making design much easier. So the cubic bezier, with its nodes and control points, is the design industry's drawing standard.

Based on an article: cr8software.net/article004.html

Appendix IV - Keyboard Shortcuts

SHIFT O	Open font file
SHIFT S	Save font file
ARROW KEYS	Move selected points (use shift to move faster)
+ and -	Magnify/reduce edit window
CTRL F	Find glyph
CTRL B	Toggle preview fill
CTRL T	Toggle view mode
CTRL Y	Toggle mapping view
CTRL H	Toggle glyph preview panel
CTRL G	Show-hide grid
SHIFT G	Toggle snap-to grid
SHIFT CTRL G	Set grid size
CTRL W	Show-hide guidelines
SHIFT W	Toggle snap-to guidelines
SHIFT CTRL W	Set guidelines
CTRL X	Copy whole glyph
CTRL C	Copy selected points
CTRL V	Paste
SHIFT CTRL V	Paste metrics only
CTRL Z	Undo
SHIFT CTRL Z	Redo
CTRL U	Glyph information
CTRL A	Select all
CTRL D	Select none
ESC or Enter	Deselect point
CTRL N	Create a new glyph
CTRL M	Clear current glyph
CTRL P	Point/node properties
CTRL E	Select contour
1 to 7	User-defined actions
8	Unlinked corner curve
9	Smooth curve
0	Symmetric curve
CTRL R	Reverse contour
SHIFT CTRL D	Delete contour
DEL	Delete selected points
F1 to F6	Select tools (1-6)
SHIFT F1 to F4	Select tools (7-10)

This End User License Agreement accompanies the **Type light 3.2 font editor** ("Software") and applies to all associated files (except third party fonts), any upgrades, documentation, modified versions or updates of the Software.

1. USE OF THE SOFTWARE.
You may use the Software product for the production of fonts for personal or limited commercial use (you can sell fonts produced by the software if you are not a business, and selling fonts is not your primary income). The Software may not be used in any other commericial capacity (including the production of fonts for sale by a foundry or editing or altering of existing fonts for printing, design or other industrial or commercial use).

2. COPYRIGHT.
The Software is copyright of the Software Publisher. You may not copy, sell or redistribute the Software. This Agreement does not grant you any intellectual property rights in The Software.

3. RESTRICTIONS.
You agree not to modify, adapt, translate, reverse engineer, decompile, disassemble or otherwise attempt to discover the source code of the Software. You may not alter or modify the installer program or create a new installer for the Software.

4. LIMITED WARRANTY
In no event will the Software Publisher be liable for indirect, special, incidental, tort, economic, cover or consequential damages arising out of the use of or inability to use the Software, including, without limitation, damages or costs relating to the loss of profits, business, goodwill, data or computer programs, even if advised of the possibility of such damages. In no case shall the Software Publisher be liable for money damages exceed the amount paid by you for the Software out of which such claim arose. The Software Publisher limits liability, according to the terms of this Agreement, to the extent permissible at law.

5. THIRD PARTY COMPONENTS
This End User License Agreement does not apply to certain third party fonts bundled with this package. These fonts, and their respective licenses, are contained in the /enc subdirectory of the install directory.

The FreeType library (freetype6.dll) is used and distributed under the FreeType Project License. See freetype-license.txt in the install directory.

YOUR ACCEPTANCE OF THE FOREGOING AGREEMENT WAS INDICATED DURING INSTALLATION.

22 Barnea Circle, Auckland 0602, New Zealand.
cr8software.net

30091950