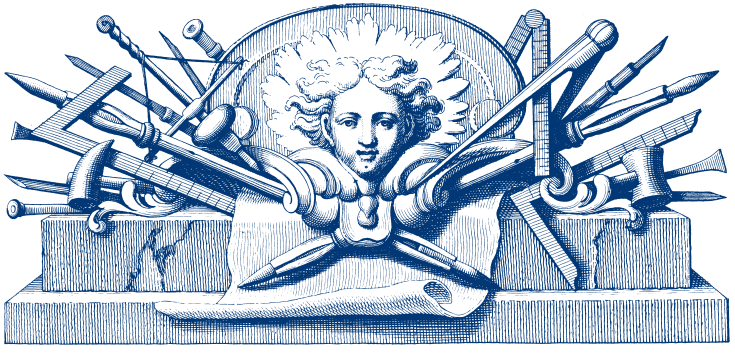
Dutch Type Library

DTL FontMaster



FontMaster™ *Utilities*

’s-Hertogenbosch / Hamburg
Summer 2004

Typography means more than bringing order to the passing on of information; it means elevating to the sublime the mould in which the process of passing on is cast.

Frank E. Blokland

Limited user rights

You may never use the software to edit data, including but not limited to fonts of which you do not own the rights, including but not limited to intellectual property rights, copyright and rights to trademark, unless the rightful claimant has given his written and signed consent.

DTL DATAMASTER

DTL IKARUSMASTER

DTL KERNMASTER

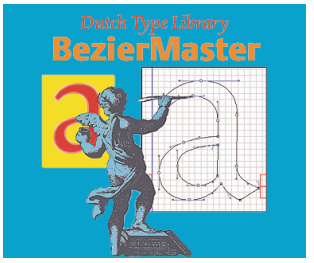
DTL TRACEMASTER

APPENDICES

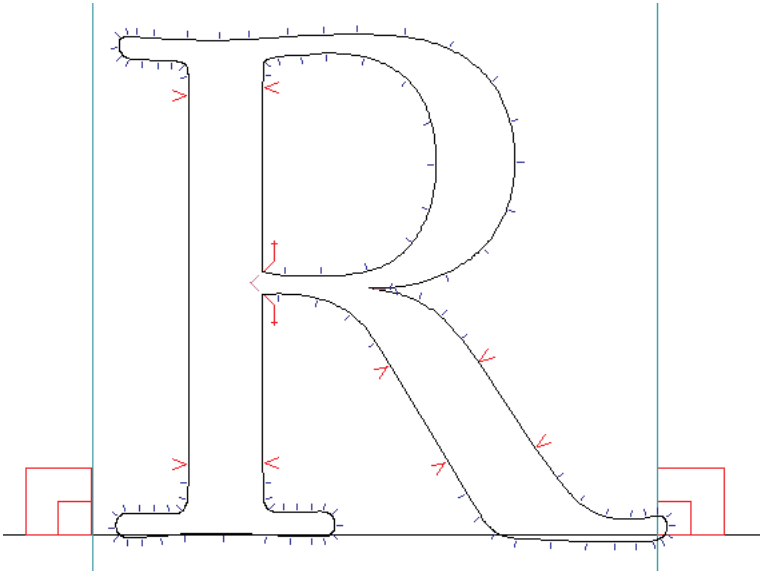
* * *

FontMaster™ Utilities

DTL FontMaster is a set of utilities for professional font production. It includes a module for designing and editing letters in bezier format (BezierMaster), interpolating fonts (BlendMaster), testing and correcting contours (ContourMaster), generating and converting font formats (DataMaster), the editing of the IK format (IkarusMaster), generating and editing kerning pairs (KernMaster) and scanning and auto-tracing letters and logos (TraceMaster). Batch functions enable you to process large quantities of data at once. KernMaster is not available yet.



The modular system makes DTL FontMaster very versatile not only for users but also programmers. Because each utility can be obtained separately and functions independently of the other modules, users can define their own set depending on their production methods. The modular system makes DTL FontMaster also easy to update; the technical functionality of each module can be enhanced without influencing any of the other modules. This guarantees a stable environment for font production that is always up to date.



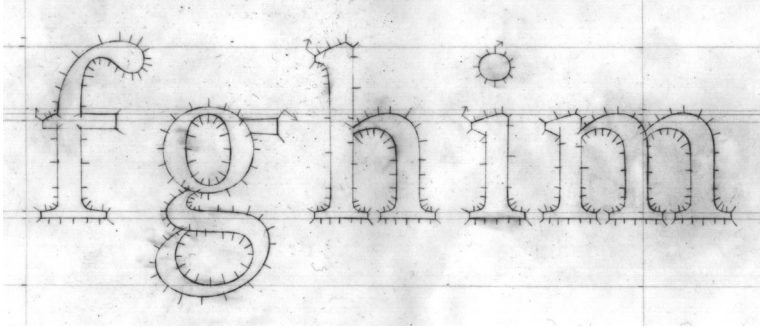
The up-to-date status of DTL FontMaster is underlined by, for instance, the way DTL DataMaster handles the production of OpenType fonts. Features can be generated by DataMaster based on the available characters in the font database. This way all the meticulous work needed for the production of OpenType is done automatically by the program. Because the OpenType functionality is completely based on the Adobe SDK, the result is fully compatible with Adobe's OpenType Pro fonts.

FontMaster™ Utilities

The Dutch Type Library has produced digital typefaces since the end of the 1980s. As font production at DTL became more professional and complex, the need for specialist font software was grew In the second half of the 1990s, the programmers of URW++ /DTL Germany started programming the modules for digital font production, which run under Windows and Mac OS. In the course of time, the DTL FontMaster team further elaborated upon the functionality. The software that was developed for the Dutch Type Library is now available to everyone.

DTL FontMaster was developed using the Dutch Type Library's more than ten years of experience in the field of font production. In 1990, the Dutch Type Library started supplying digital letter types after years of preparation, which resulted in top-quality typefaces such as DTL Argo, DTL Documenta, DTL Fleischmann, DTL Haarlemmer and many others. The company has since grown into the largest of its kind in the Netherlands.

Attention is focused primarily on contemporary designs in the production of typefaces, but DTL also revives valuable historic typefaces. The Dutch Type Library feels bound to the rich Dutch typographic tradition and DTL's collection can be compared with the very best that the famous type-foundries have produced in the past.



(left) Artwork of DTL Fell

The Dutch Type Library was commissioned to produce company typefaces for the European Union, Museum Boijmans Van Beuningen Rotterdam and The Hague's City Museum. Moreover, DTL supplied the company letters to the New York Stock Exchange, Germany's Phoenix Television Broadcasting Company, Amnesty International USA, Emerson, F. van Lanschot Bankiers, Amsterdam Historic Museum, the Dutch Open-air Museum, Museum Plantin-Moretus in Antwerp, the Royal Library in Brussels, the Royal Museum for Central Africa (Tervuren, Belgium), Oxford University Press, Academisch Ziekenhuis Groningen (Groningen Teaching Hospital), Koninklijke BAM/NMB Groep, Heerema Group, Radio Nederland Wereldopmroep (World Service), Danish Railways (DSB), Agis Groep (health insurer), Finland's most popular newspaper *Helsingin Sonamat* and banks and museums all over Europe.

Currently the following fifteen font families produced by the Dutch Type Library are available:
–DTL Albertina
–DTL Argo
–DTL Caspari
–DTL Documenta & Sans
–DTL Dorian
–DTL Elzevir
–DTL Fleischmann
–DTL Fell
–DTL Haarlemmer & Sans
–DTL Nobel
–DTL Paradox
–DTL Prokyon
–DTL Romulus
–DTL Unico
–DTL VandenKeere

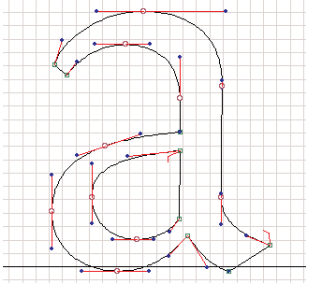





The DTL FontMaster Team from top left to bottom right:
Frank E. Blokland
Gu Jun
Hartmut Schwarz
Peter Rosenfeld
Axel Stoltenberg
Dr. Jürgen Willrodt

The DTL FontMaster Team comprises experts in the field of font production. The programmers are: Gu Jun, who is very experienced in programming in C++ and MFC and has been working on Ikarus for Windows since 1996; Hartmut Schwarz, one of the developers of Ikarus M; Axel Stoltenberg, Ikarus programmer since 1984; and Dr. Jürgen Willrodt, who has been associated with URW as a programmer since the 1970s. Peter Rosenfeld, as the Director of DTL Germany is responsible for coordination in Hamburg, where the programmers work. Frank E. Blokland, who took the initiative to develop DTL FontMaster, is the project manager and designer.

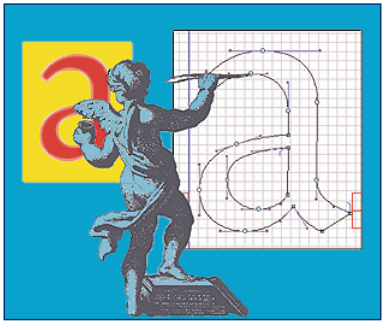
For more information visit: www.fontmaster.nl



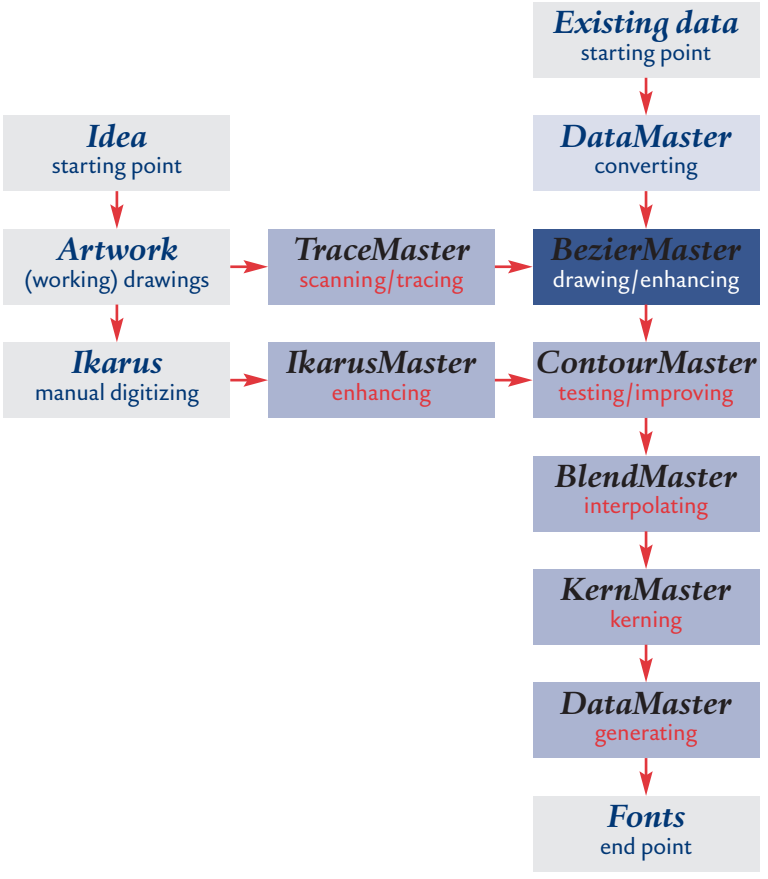
Engraving by Cornelis van Noorde (dated 1769) of Johann Michael Fleischmann (1707–1768), one of the greatest and most famous punch cutters who worked most of his life in the Netherlands. The Dutch Type Library produced a revival based on his work. (DTL collection)

Dutch Type Library

DTL BezierMaster



's-Hertogenbosch / Hamburg
Summer 2004



The diagram shows a typical workflow based on the modules of DTL *FontMaster.*

DTL BezierMaster is an editor for outlines in bezier format. The program is intended for drawing and managing character sets, but can also be used, for example, to check the corrections made in DTL ContourMaster via an optical check. The interface can be changed according to one's preference: e.g. the colour display of the contour and bezier check points (BCPs) can be adjusted, and the resolution-sensitivity of the cursor is fully verifiable. DTL BezierMaster has many unique properties, such as the option to check the BCPs with the arrow keys and the option to load the same font into the foreground and background. Changes that are made in the foreground are simultaneously shown in the background. In this way, increasingly refined corrections can be made to a letter. The Font Administration tool in DTL BezierMaster makes managing code pages surveyable, which is particularly necessary for the production of large Unicode-based fonts, such as OpenTypes.

User Interface Guidelines

The user interface of the modules of DTL FontMaster follow the standards of Mac OS and Windows.

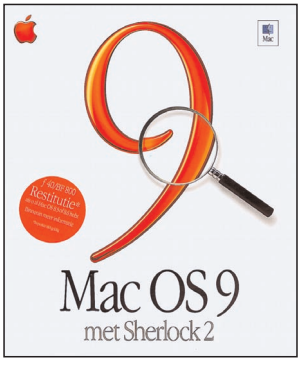
The tool bar showing the functions as graphic symbols can be moved around with the mouse to a desired place.

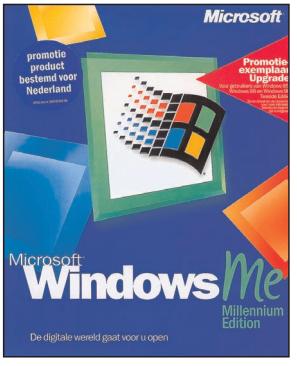
Function selection works as usual: if you click the mouse inside the menu bars, for example **File**, you can select the functions in 'pull down menus' which are displayed on the screen. Select the requested function, for example *Open …*, inside the menu by selecting the item with the mouse. For some functions shortcuts on the keyboard are also available. They are displayed as shown at the right and can be executed without pulling down the **File** menu first.

Under Windows the combination for *New* for example is <Ctrl> + N, on the Macintosh the key is ⌘ + N.

On Windows systems you can also select any function by pressing the <Alt> key and the underlined character shown in the pulldown menu.

A summery of the shortcuts for Mac OS and Windows for DTL BezierMaster is given at the end of this chapter.

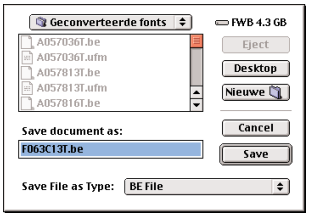




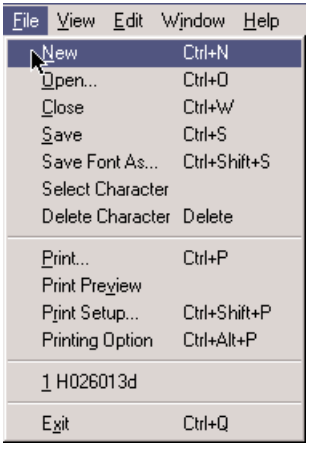
Nomenclature

In this manual as well as in the modules we do not distinguish between glyphs and characters. The term *character* is used many times to indicate an outline description, although a character actually can be represented by different glyphs. For example the character 'a' can look like this:

a*a*·a*a*·a*a*



The screen dumps in this manual were made with alternately the Mac OS and Windows versions of DTL FontMaster.



Shortcuts are displayed in the pull down menus next to the related function.

DTL FontMaster runs under Mac OS 8.6 and higher, including Mac OS X Classic mode and under Windows 95/98, ME, NT, 2000 and XP.

Outline Description

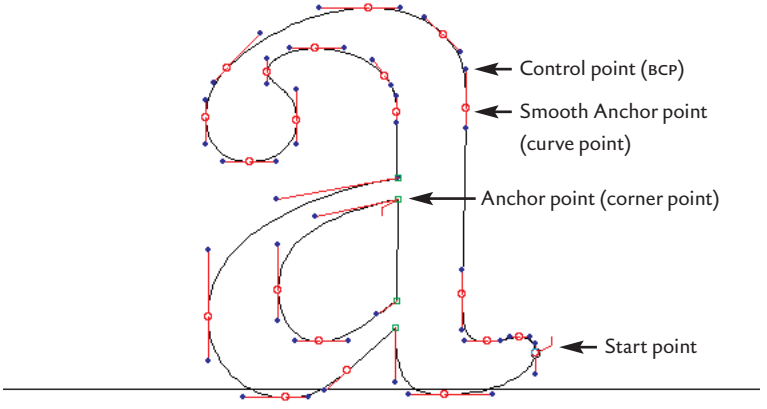
The outlines of the glyphs in the font are described by Bezier curves. The Bezier format describes an outline by either straight lines or a Bezier curve section, which is a third-degree polynomial.

A Bezier section is completely determined by two anchor points and two control points. Anchor points are the start and end of a Bezier section and are on curve whereas Control points are off curve. The slope (tangent direction) at the end and beginning of a Bezier curve is given by the direction from the anchor to the control point.

Since the Bezier sections are completely locally determined, it can not be guaranteed that two joining Bezier sections have the same slope at the junction point. Usually this will not be the case, although the design of characters does mostly require smooth transitions between adjacent Bezier sections. This can be achieved if the anchor point and the two neighbouring control points of the two different Bezier sections are aligned on a straight line. In order to aid the type designer in the design of smooth and tangentially continous outlines, four different type of point are available:

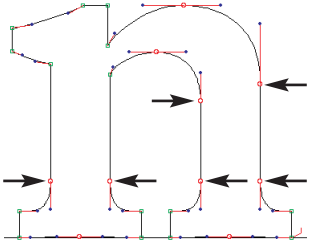
NOTE: The BE file format is considered as a database format. Currently one file can contain up to 22000 glyphs. Glyphs are identified by 2-byte numbers (1–65535). The data consists of header, contour header and outline. There is no hinting information in the data. For PostScript Type 1, TrueType and OpenType production additional information is necessary and has to be supplied via text files, like UFM, AFM, etc. (more info about these text files can de found in the appendices).

The format specification is public and can be found in Dr. Peter Karow's Schrifttechnologie (Berlin, 1992).



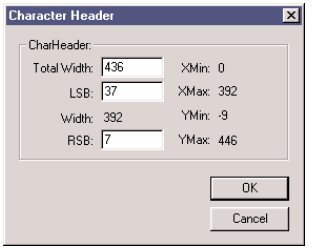
TIP: If you use <Ctrl> + mouse click the program will change the label from Anchor to Smooth Anchor point and vise versa. Using the Change Label function from the Function Tool Bar makes it possible to change straight lines into curves and vice versa, this way adding and removing Control points.

1. **Start points**: only one start point for each contour is allowed. Start points are marked in red by default . The start point should be an anchor point. The hooked line shows the direction of the contour.
2. **Control points**: off curve points, by default marked in dark blue. These are also called Bezier Control Points (BCPs).
3. **Anchor points**: also called corner points: these points are on curve points, marking the end or beginning of a straight line or a Bezier curve. They are marked in green by default.
4. **Smooth Anchor points**: These points are on curve points, marking the end or beginning of a straight line or a Bezier curve. Additional to normal anchor points they also force tangent continuity between the adjacent sections. They are marked in red by default.

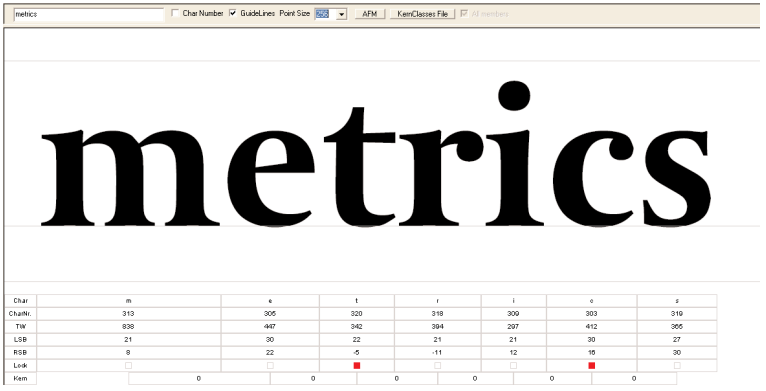


Smooth Anchor points are used in this example as tangent points, this way forcing tangent continuity between the adjacent sections.

The available functions of DTL BezierMaster can operate on three different 'objects':
– Points as described above
– Contours. Contours consist of several Bezier sections and/or straight lines. They should be closed.
– Characters. These consist of zero, one, or several contours.
There is additional information for the character such as the width and the left and right sidebearings. These values can not be edited interactively in the Character Edit Window, but can be changed numerically in the *Character Header* from the **Edit** menu or in the ***Metrics Window***.



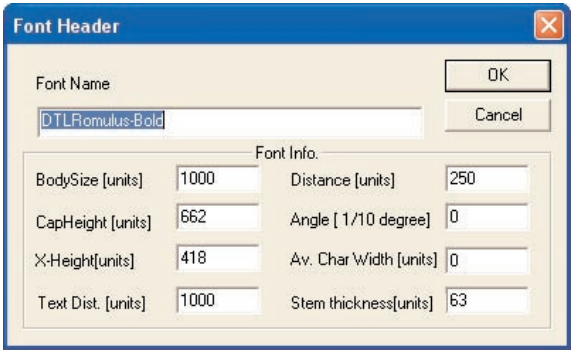
*The character width can be edited with the Change Character Header function from the **Edit** menu.*



In the Metrics Window the character width can be altered.

The general information about the complete font, such as the Bodysize, is contained in the *Font Header*. For some parameter values it is important to know the bodysize of the font, which usually is 1000 (in a PostScript font), 2048 (in a TrueType font) or 15000 (in standard Ikarus data). Font Header information can be viewed and changed with *Change Font Header* function in the **Edit** menu.

*The Font Header contains general information about the complete font. These settings are used for instance for the V/H Guide Lines function in the **View** menu and for the generation of the UFM file.*



Selecting points

DTL BezierMaster uses a standard mouse with one button on the Mac or the left button of the mouse under Windows.

For some functions it is necessary to click a mouse button and at the same time press down the <⇧Shift> key or the <Ctrl> key. On the Mac the command <⌘> key is used.

These combinations will be referred to as <⇧Shift>-mouse button, <Ctrl>-mouse button or <⌘>-mouse button.

The *arrow* tool (–>), also called *pointer* tool, is the default function for selecting objects like points, contours or complete characters. Clicking the mouse button near an outline point will select this point.

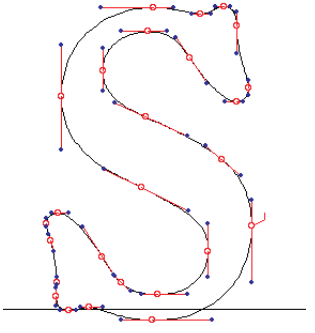
The selection will be constrained to a certain radius around the point. This radius can be modified by the user in the **Config menu**. Clicking outside this radius will deselect all selected points.

If no point is selected while a function is executed the whole character will be modified by default.

Arrow or pointer tool

1. Selecting single points

To move or delete for one point you must click near the point on the screen to select it, in other words, select by clicking.



No points selected

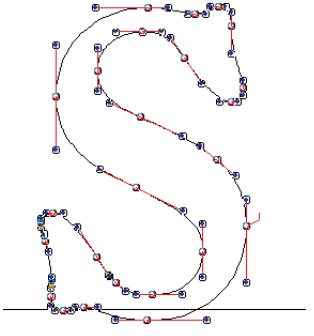
2. Selecting multiple points

2.1 *Collecting points*

Select the first point by clicking on it. While holding down the <⇧Shift> key, click again on another BE point to add it to the selection. You can also deselect an already selected point by clicking on it while holding the <⇧Shift> key.

2.2 *Window-in Selection*

You can select all points within a user-defined window. To do so, click away from an outline point, hold down the mouse button and drag the cursor over the points to be selected. You can add more points to the already selected collection by pressing the <⇧Shift> key and repeating the window selection. The new points will be added to the already selected ones.



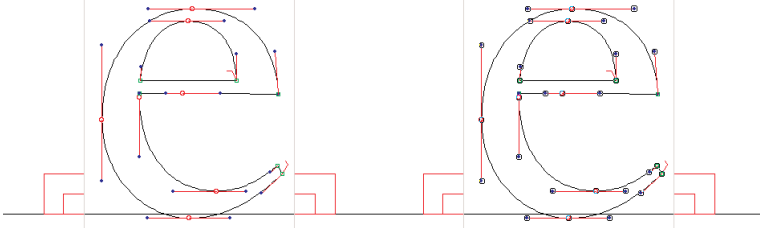
All points selected

3. Contour Selection

To select a contour, double click near an outline point off the requested contour. To select additional contours click again near an outline point of another contour while holding down the <⇧Shift> key. To deselect a contour click near an outline point off the requested contour while pressing the <⇧Shift> key.

4. Character selection

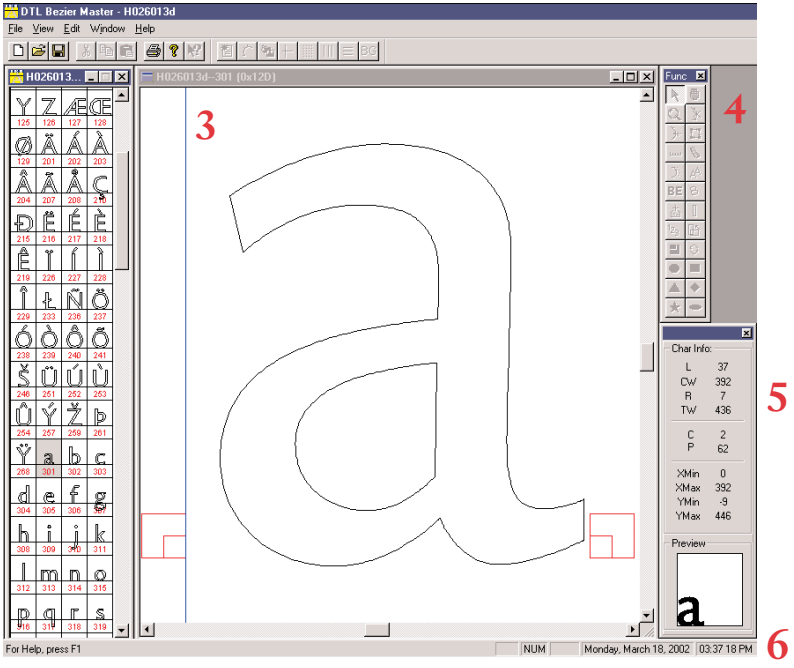
To select all points of a character use the standard shortcut <Ctrl> + A. Some of the more advanced functions, like *Hidden Lines*, assume that all points are to be processed if none is selected.

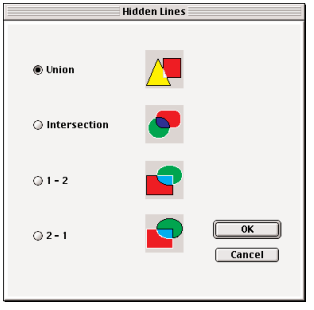


A glyph can also be selected by double clicking while positioning the pointer tool inside the contour.

Screen Layout

The DTL BezierMaster editor has the following components in the screen layout:





*The Hidden Lines function from the **Special** menu.*

1. Tool Bar
2. Character List Window
3. Character Edit Window (several windows are possible)
4. Function Tool Bar
5. Character Display Info and Preview
6. Status Bar

Menu functions

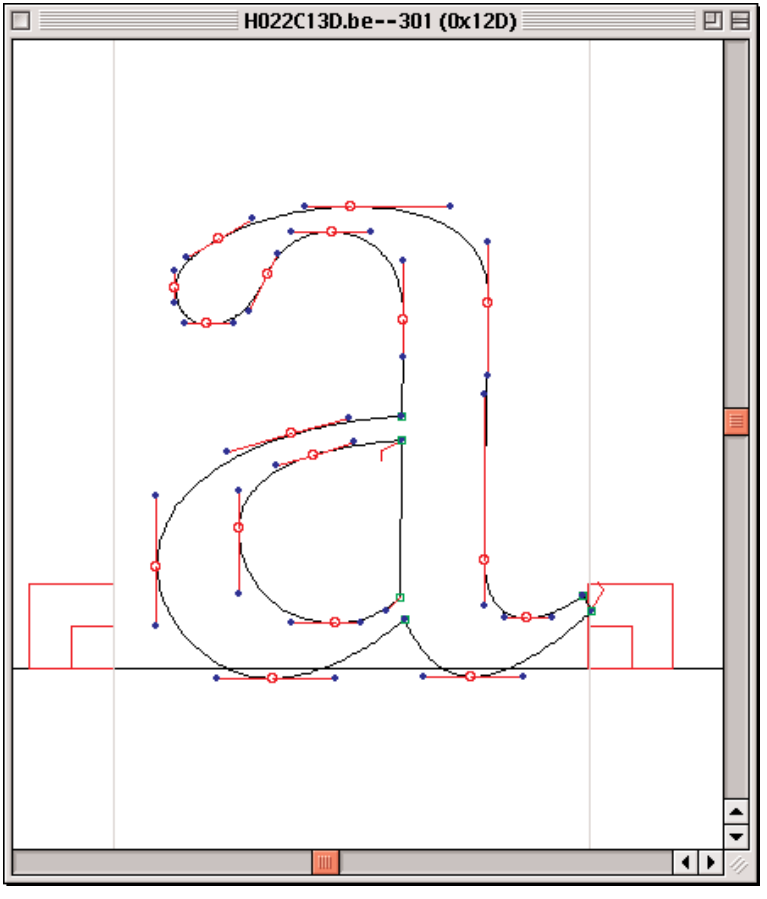
The menu functions are selected from the pulldown menus or with the defined shortcuts from the keyboard.

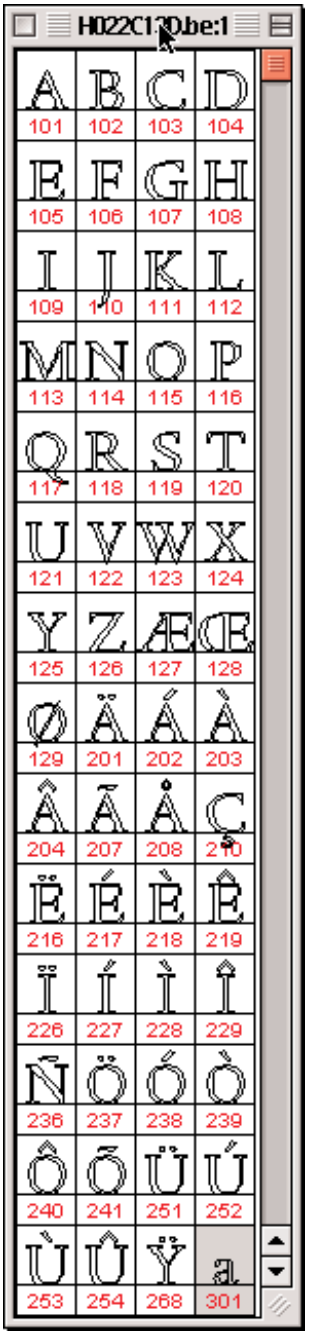
The following menus are available while the Character List Window (right) is active:

File View Edit Window Help

In the Character Edit Window the actual designing takes place.The following menus are available while the Character Edit Window (below) is active:

File Edit View Tools Function Special Config Window Help





The Character List Window (top) shows an overview of all characters in the database. The Character Edit Window is shown left.

FILE MENU

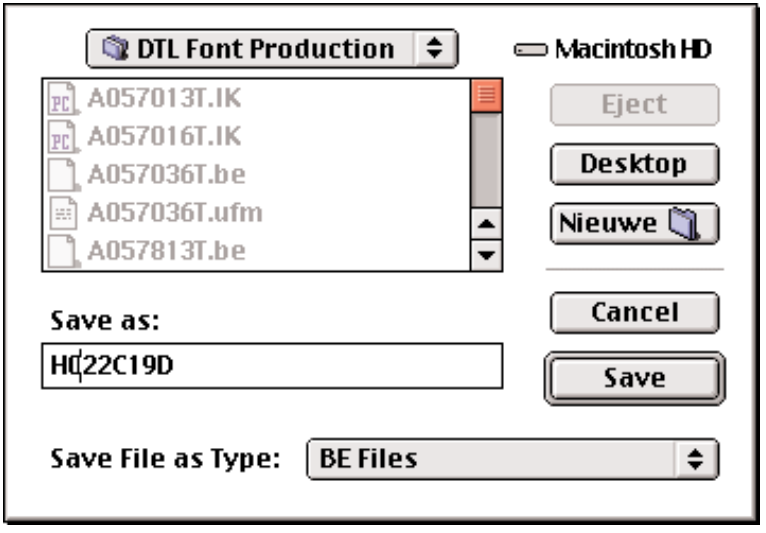
The file menu has the following functions in the pull down menu:

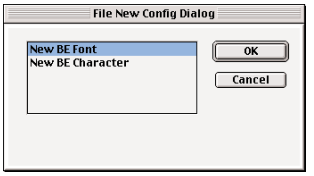
New (⌘ + N) (Ctrl + N)

This function allows you to create a new font or a new character in an already open font.

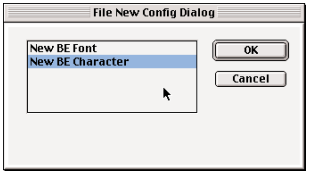
1. New font

Generates a new BE font. You should name and save the font first. Afterwards you insert the character number of the character in the new font you wish to create.





Before a new BE character can be created, first a new BE font has to be generated.

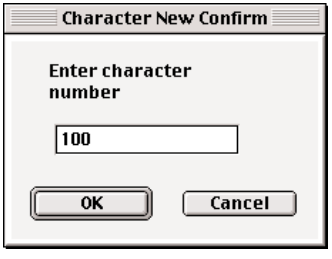


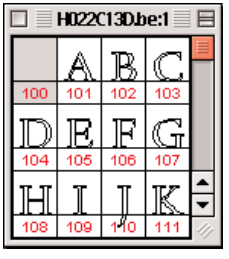
After a new BE font is generated, a new BE character can be created.

2. New Character

Insert the number of the new character you wish to insert into your currently edited font in the following dialog box.

Allowed character numbers are in the 16-bit range i.e. from 1 to 65535. 0 (zero) is not an allowed number. Character numbers can be found in Appendix IX: *Character number listing*.





After a character number has been defined, a new slot in the Character List Window is generated.

TIP: Instead of generating a new character by entering the appropiate BE number, the Font Administration tool from the View menu can be used. Double clicking with the mouse on a cell of a particular codepage will open the character and add the BE number to the database accordingly.

Open (⌘ + O) (Ctrl + O)

Opens an existing font using the standard open file dialog box. After succesfully opening the font the program will display all characters in the Character List Window. This window can be used to select characters to be edited by simply clicking into the small window. Several character edit windows can be opened simultaneously.

On Windows systems there is the option to use abbreviations (archives) for predefined directories. If a certain flag in the Windows registry is set, the program will accept only input from archives and display a special open dialog.

Close (⌘ + W) (Ctrl + W)

This function works differently depending on which window is currently active. If the character list window is active, it will close the font which you are working on. If you have changed some characters, you will be asked whether to save or not save the changes you made. All open character edit windows will be closed too. If the character edit window is active only this particular window will be closed.

Save (⌘ + S) (Ctrl + S)

Saves the data to disk. It is recommend to use this function frequently to avoid a loss of data. This function works both in the Character List Window and the Character Edit Window.

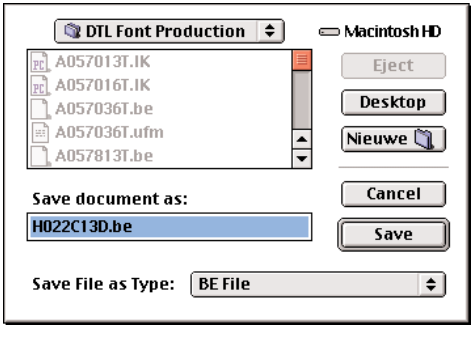
Save as (⌘ + ⇧Shift + S) (Ctrl + ⇧Shift + S)

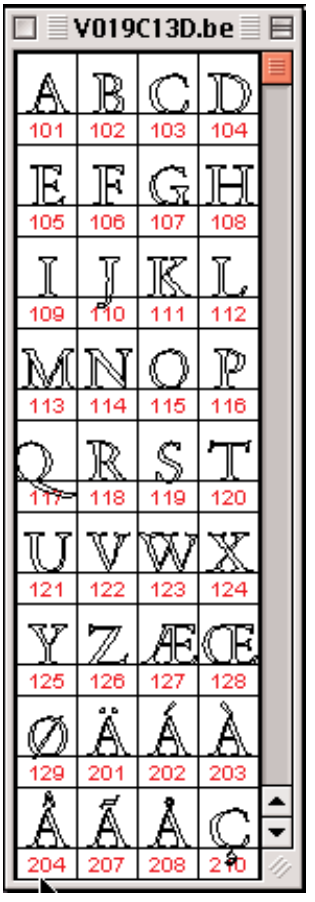
This option functions different depending on the active window:

1. Character Edit Window active

Saves the character you are working on to a different character number in the font database. The Character Edit Window will subsequently contain the new character. The previously edited character will be closed without saving the changes.

This function can be used to duplicate characters to other positions, as a basis to construct new characters, such as accents, from existing ones, or simply to reposition characters.





After succesfully opening the font the program will display all characters in the Character List Window.

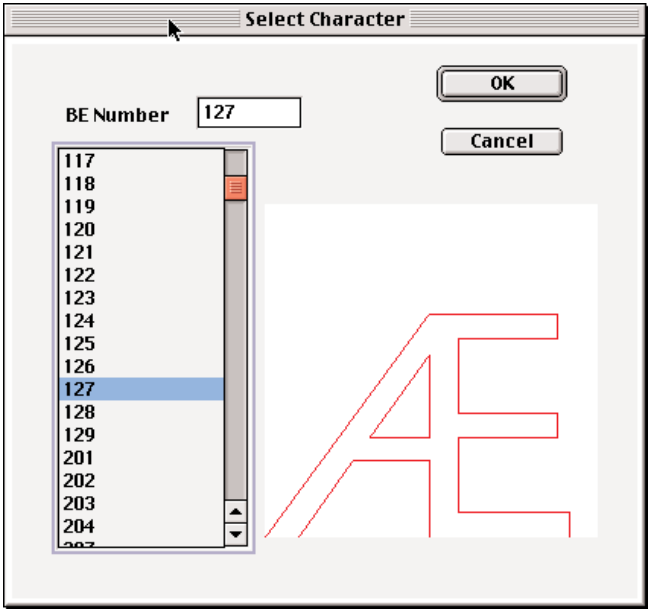
The Character List Window has to be active for saving the font under a different name.

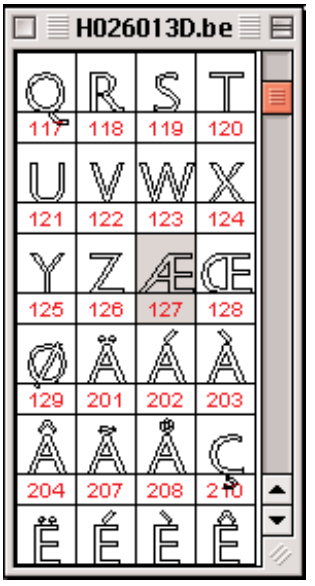
2. Character List Window active

With this function you can save the font under a new name.

Select character

In the following dialog box you can select a character by double clicking its number. The image of the character will be displayed as you click on the number once. You can also enter a character number numerically or use the arrow keys (up and down) to scroll through the character list. Double clicking on a character in the Character List Window gives the same result.





The numbers shown in the Select Character dialog are, of course, the same as in the Character List Window.

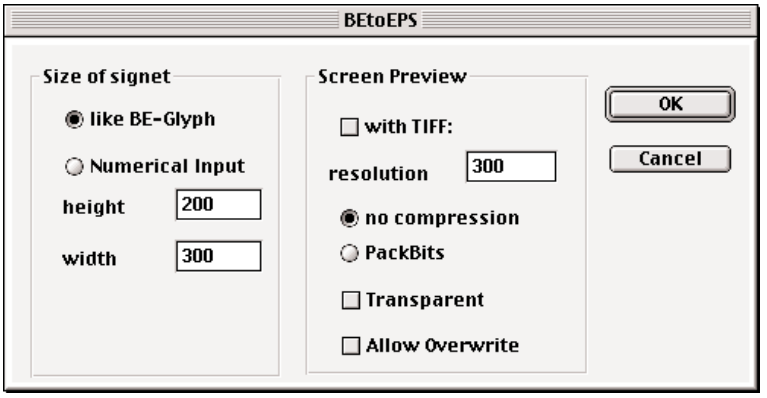
Delete character (← Backspace) (Delete)

To delete a character, just select the character in the Character List Window on the left side by clicking once on the character and then press the <←Backspace> or <Delete> key.

Import EPS file

Reads an EPS file and converts it to BE format. For instance EPS files made with Adobe Illustrator® can be imported. The EPS data will be converted to Bezier format and merged into the currently active Character Edit Window.

The size and positioning of the BE data is taken from the EPS data and scaled to the bodysize of the font. Please note that the EPS file must have the suffix *.eps* to be recognized. The EPS file must also not contain multiple layers; these have to be removed first before importing the data in the Character Edit Window.



The BE to EPS dialog offers a number of options for the export of EPS files.

EPS Output

Glyphs can be exported individually as EPS data. The same functionality is available for groups of glyphs from the *EPS Output* option in the **Batch** menu.

TIP: To export BE glyphs as EPS data in groups, the EPS Output function in the Batch menu can be used.

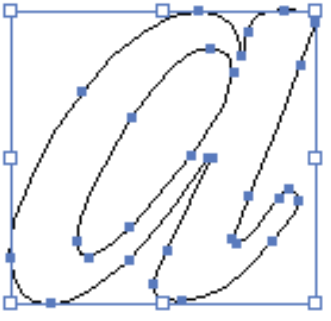
– Size of signet

The glyph(s) can be exported with unchanged size by selecting the option *like BE-Glyph*. This preserves the original relation to the other glyphs in the font database, which makes re-importing (for instance after editing in Adobe Illustrator®) possible without any scaling. With the option *Numerical Input* the bounding box can be scaled to any size measured in millimeters.

– Screen Preview

The EPS files can be provided with a TIFF for screen previewing. As an option the required resolution can be entered. The default reolution is 300 dpi. The TIFF files can be compressed with Apple's *PackBits* format or leaved uncompressed, which is the default setting.

Further options are *Transparant* and *Allow Overwrite*, which let the program overwrite EPS files with the same name. The name of the EPS is taken from the name of the database plus the Character Number. Exporting the glyph with Character Number 302 of the H022013D named database will result in H022013D_00302.EPS. The EPS files are automatically stored in the directory that contains the used font database.



Exported EPS files can be imported for instance in Adobe Illustrator®.

Print (⌘ + P) (Ctrl + P)

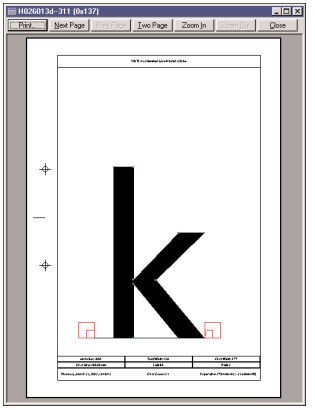
Uses the standard printer driver print dialog. Select the number of pages you wish to print or set the properties of the printer. These depend on the printer you have installed.

Print Setup ... (⌘ + ⇧Shift + P) (Ctrl + ⇧Shift + P)

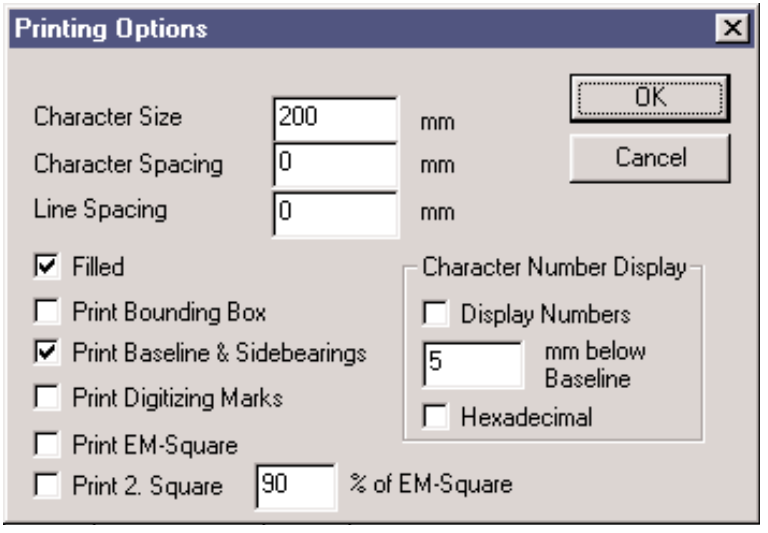
This function allows you to change the printer and printer properties.

Print Preview

This function works differently depending on which window is active.
–If the Character Edit Window is active, a printout of this character will be generated with the selected print options and be displayed in the currently active window as a preview.
–If the Character List Window is active all characters or the selected ones (see **File Menu** –> *Printing Options* –> *Text Options*) will be prepared for printing and the generated page will be shown in the character list window. Since this window is quite narrow it should be enlarged to show the complete page and all the options for the display of the preview.



Print previewing is possible.



A large range of print options is available when the Character Edit Windows is active.

Print Options (⌘ + ⌥Alt + P) (Ctrl + Alt + P)

These options are different depending again on the active window:

1. Character Edit Window active

This function allows you to set different options for proofing of individual characters.

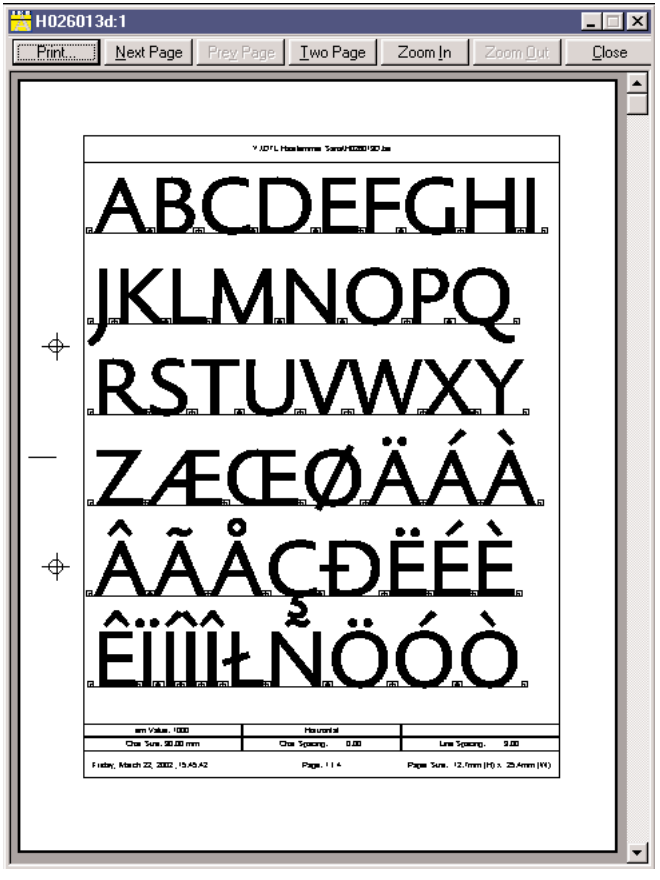
1.1 *Character Size* (bodysize in millimeters).
1.2 *Character Spacing* (spacing in millimeters).
1.3 *Line Spacing* (line spacing in millimeters).
1.4 *Filled* (displays the character solid or as outline).
1.5 *Print Bounding Box* (shows the character bounding box).
1.6 *Print Baseline & Sidebearings* (shows baseline and sidebearings).
1.7 *Print Digitizing Marks* (shows anchor and control points).
1.8 *Print EM-Square* (shows the EM around the character).

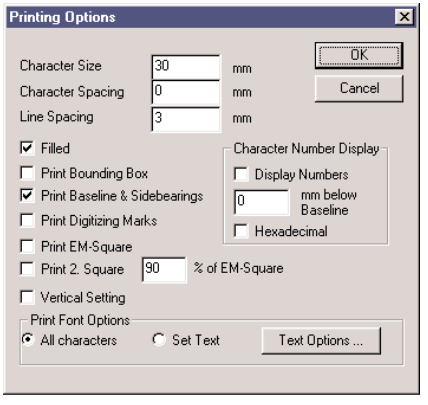
1.9 *Print 2. Square* (shows the facesize for Kanji).
2.0 *Display Numbers* (shows the character number at the set distance below the character).
2.1 *Hexadecimal* (Output of character number in hexadecimal; decimal is the default setting).

2. Character List Window active

This function allows to set different options for proofing of several or all characters. Additional options compared to the printing of a single character are:

2.1 Character spacing (distance between the sidebearings of two characters).
2.2 Line Spacing (distance between the two lines of text).
2.3 Horizontal/vertical setting (option for Kanji setting).

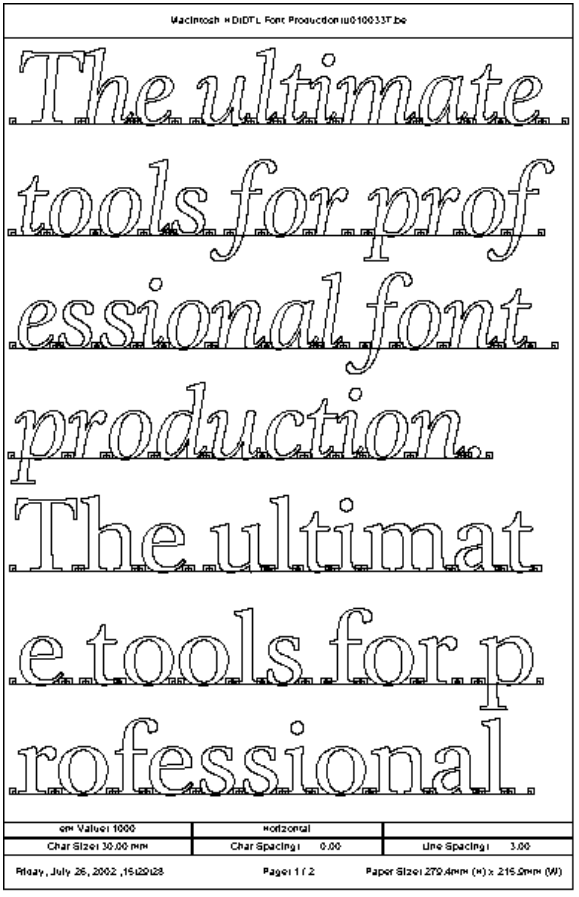


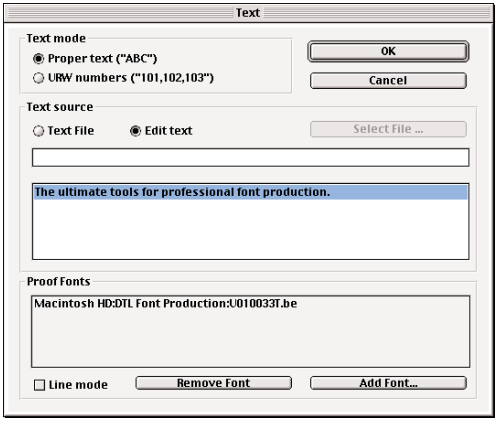


A large range of print options are available when the Character List Window is active. The options selected in the dialog above result in the output shown on the left.

2.4 TextOptions. You will see a dialog which allows the input of character numbers or proof text. For proof text you can select an existing text file or edit text in the Edit Window. You can also choose different fonts for printing the selected text.

After selecting Text Options … in the Printing Options dialog it is possible to enter a text for proofing the font(s), as shown at the left.





(left) In case you have chosen more than one font to display the text, you can display the whole text in the first font, then in the second, the third, etcetera. Note that the line mode will change the Proof fonts after each line of text, not after the complete text.

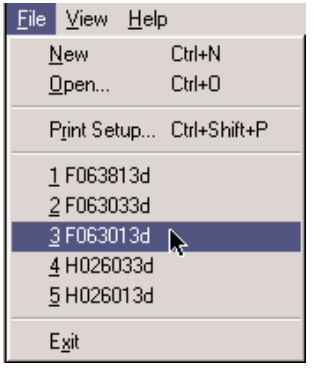
File list, 1 … 2 … 3 … 4 …

Select a font from up to eight font files listed in the **File** menu. These fonts have been opened before and have been memorized by the program automatically.

Exit (⌘ + Q) (Ctrl + Q)

Exit the program. If unsaved data is still in memory, the program will ask you if you would like to save these changes to disk.

On Windows you can also use the standard <Alt> + <F4> to close the program.



The File List, 1 …, 2 …, 3 …, 4 …, function will show up to eight of the most recent used files.

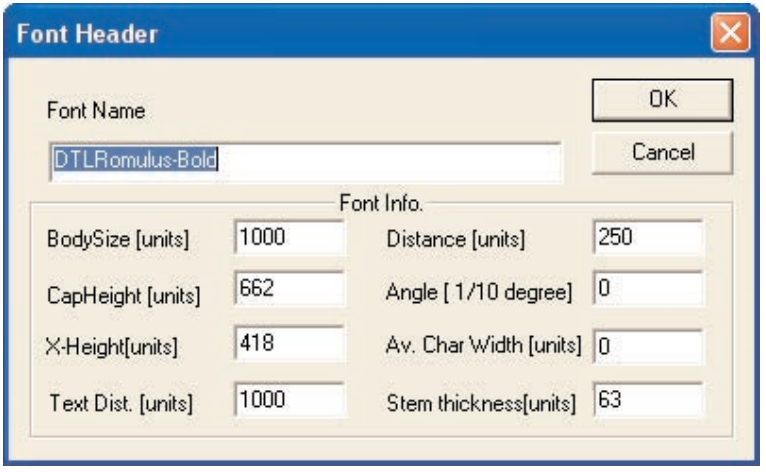
EDIT MENU

These options are different depending on the active window:

1. **Character List Window active**

Change Font Header (⌘ + ⇧Shift + F) (Ctrl + ⇧Shift + F)

Use this function to numerically change the Font Header as well as to display the content. All values are font specific and might be used for further format conversions.



The font header contains general information about the complete font. These settings are used for instance for the V|H Guide Lines function in the ***View*** *menu and for the generation of the UFM file.*

The *Distance* is the space between the baseline descender line. The BodySize is normally the distance between the extremes of the ascender and the descender.

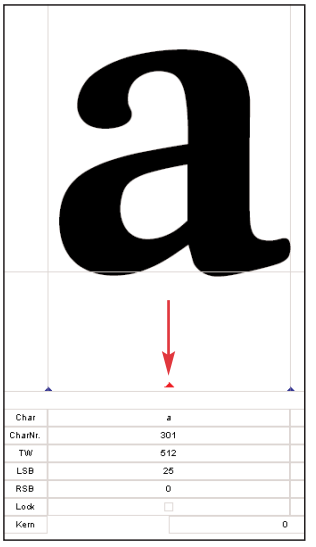
Metrics Editor

This is a very powerful tool to adjust the positions and widths of characters. The Metrics Window can be resized to take full advantage of the screen resolution. The anti-aliased shown Characters can be selected by keystrokes or by character (database) number. The size of the characters can be defined by point size.

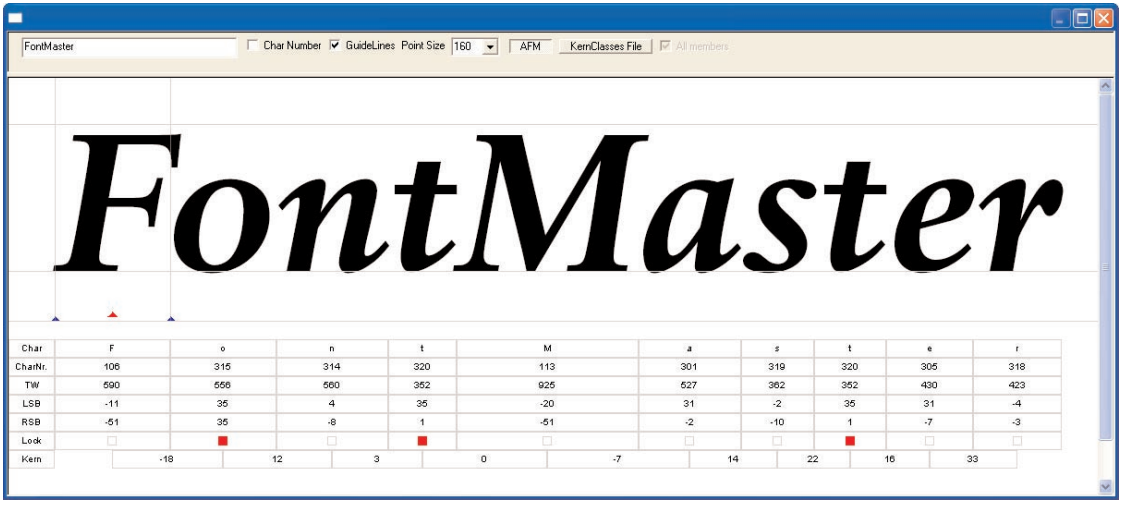
The position of the characters can be changed by selecting and dragging the triangle below the line that marks the *Distance* value in the Font Header (normally the length of the descenders). The triangle marks the centre of the *bounding box*. By default the triangle is blue but in case a character is selected the triangle in the centre is red. Only a selected triangle can be moved. The side bearings can be changed by dragging the blue triangles (these don't change colour) that mark the widths of the characters. Always take note of the fact that moving the side bearings only affects the width of the selected character (marked with the red triangle in the center). The side bearings are shown if the option *guidelines* is selected.

Changes to the width can also be made numerically by altering the values of the side bearings indicated by LSB (left) RSB (right). The changes made in the Metrics Editor are saved automatically to disk. There is of course an undo function. To prevent any errors, there is the possibility to lock the position and the width of the shown characters.

Further options include the import and export from kerning information from AFM file or kern feature file, which for instance can be generated both by DTL KernMaster.

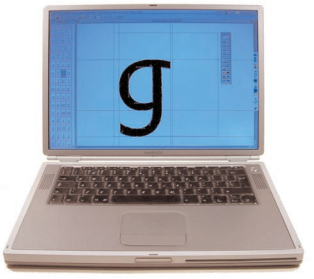


A red triangle in the centre (of the bounding box) marks a selected character.



The Metrics Editor is a powerful tool to adjust the positions and widths of characters.

2. Character Edit Window active



Undo (⌘ + Z) (Ctrl + Z)
Undoes the last editing action. The program supports up to 50 undo levels.

Redo (⌘ + Y) (Ctrl + Y)
Redoes the last editing action with the **Undo** function. Redo supports up to 20 different redo steps.

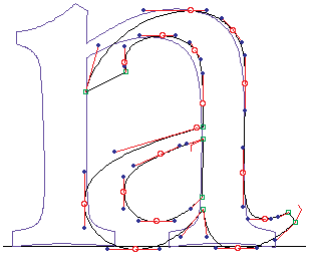
Undo Character Editing
Undoes all changes since the last **Save**.
The program will ask for a confirmation before all edits are discarded.

Cut (⌘ + X) (Ctrl + X)
Deletes and simultaneously copies one or more selected contours to the clipboard.

Copy (⌘ + C) (Ctrl + C)
Copies all selected contours to the clipboard.

Paste (⌘ + V) (Ctrl + V)
Pastes the content of the clipboard into the currently edited character at the position which is defined as the original position plus a fixed offset in X and Y which is currently set to 20,20 for 1000 EM.

Paste (⌘ + ⇧Shift + V) (Ctrl + ⇧Shift + V)
Pastes the content of the clipboard into the currently edited character without offset.



Copying in fore- and background can be done using shortcuts.

Select all points (⌘ + A) (Ctrl + A)
Selects all points of the character.

Copy into Background (⌘ + Ctrl + C) (Ctrl + Alt + C)
Copies the character in the foreground into the background.

Replace by Background (⌘ + Ctrl + R) (Ctrl + Alt + R)
Replaces the character in the foreground by the character in the background.

Paste from Background (⌘ + Ctrl + V) (Ctrl + Alt + R)
Adds the character in the background to the character in the foreground.

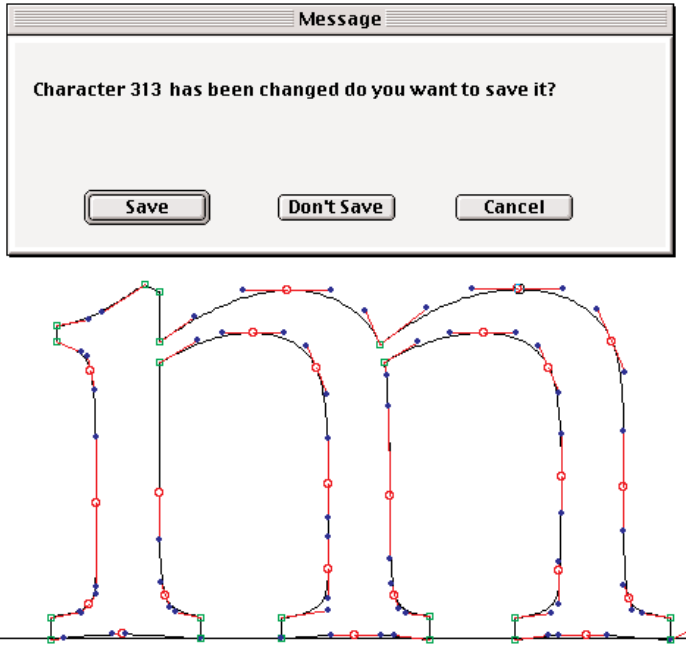
Next Character (⌘ + →KeyRight) (Ctrl + →KeyRight)
Selects the next character in the currently active character edit window

with respect to the character number. If the currently edited character has been modified, the program will ask if you want to save the edits.

If the background is active, the character in the background will be changed too if the same number as used for the foreground character exists.

Previous Character (⌘ + ←KeyLeft) (Ctrl + ←KeyLeft)
Selects the previous character in the currently active Character Edit Window with respect to the character number. If the currently edited character has been modified the program will ask whether to save the edits or not.

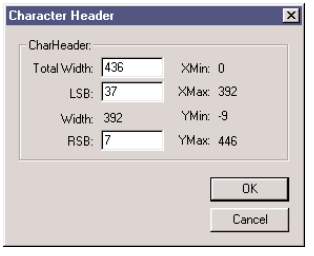
If the background is active, the character in the background will be changed too if the same number as used for the foreground character exists.



If changes have been made to the current character, the program will ask whether to save these or not before showing the next or previous character.

Change Character Header (⌘ + I) (Ctrl + I)
Use this function to numerically change the left side bearing (LSB), right side bearing (RSB) and total width (the width is calculated automatically).

You can edit the fields on the left side. You can not edit the xMin, xMax, yMin and yMax fields. These will be calculated automatically.



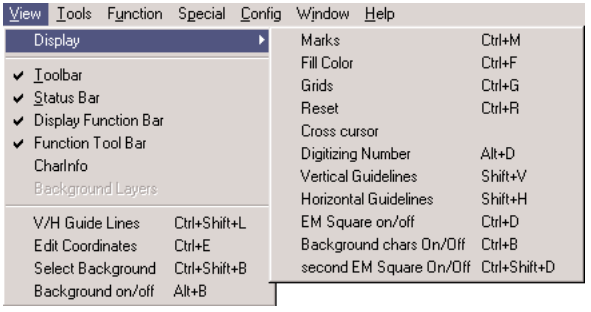
In the Character Header dialog the width and side bearings can be numerically changed.

The side bearings can also be changed by selecting and dragging in the Character Edit Window.

Change Font Header (⌘ + ⇧Shift + F) (Ctrl + ⇧Shift + F)
Use this function to numerically change the Font Header as well as to display the content. All values are font specific and may be used for further format conversions.

VIEW MENU

This menu has different options depending again on the active window. If the character edit window is active you will see the following menu entries.



Display

In this submenu several parameters for the display can be set.

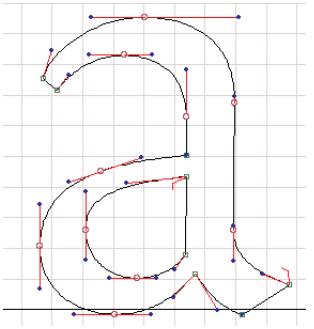
***Display Marks* (⌘ + M) (Ctrl + M)**
Displays the Bezier Anchor and Control points:
– *Start points*
– *Control points*
– *Anchor Points*
– *Smooth Anchor points*

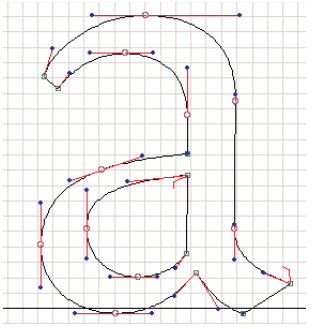
***Fill Color* (⌘ + F) (Ctrl + F)**
Use this function to switch on and off the filled display. The colour is set in the **Config Menu**: ***Editor functions and colors***.

***Grids* (⌘ + G) (Ctrl + G)**
Use this function to switch on and off a grid which is shown in the background. The grid can be defined in the **Config Menu**: ***Editor functions and colors***: ***Grid step***.

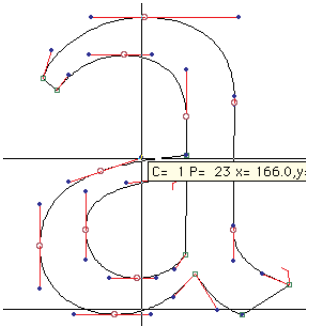
***Reset* (⌘ + R) (Ctrl + R)**
Resets the display to the default size. Use this function also to redisplay the character in case of display problems.

Cross Cursor
Use this function to switch on and off a cross-hair cursor.





The grid is defined in the ***Config Menu: Editor functions and colors: Grid step****.*



The cross cursor is an alternative for the arrow (pointer) tool.

Digitizing Number **(⌥Alt + D) (Alt + D)**
Use this function to switch on and off the digitizing numbers of the anchor and control points, corresponding to the Edit XY list.

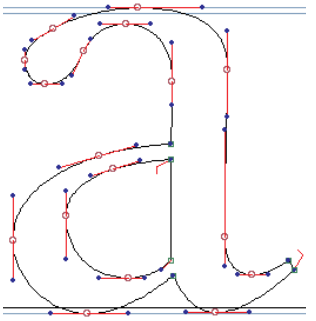
Vertical Guidelines **(⇧Shift + V) (⇧Shift + V)**
Use this function to switch on and off the vertical guidelines as set in **View** –> ***V/H Guidelines***.

Horizontal Guidelines **(⇧Shift + H) (⇧Shift + H)**
Use this function to switch on and off the horizontal guidelines as set in **View** –> ***V/H Guidelines***.

EM Square on/off **(⌘ + D) (Ctrl + D)**
Use this function to switch on and off the EM square.

Background chars on/off **(⌘ + B) (Ctrl + B)**
Use this function to hide and display the characters that are put into the background.

Second EM Square on/off **(⌘ + ⇧Shift + D) (Ctrl + ⇧Shift + D)**
Use this function to switch on and off the second EM square, as defined in the **Config** –> ***Settings*** menu.



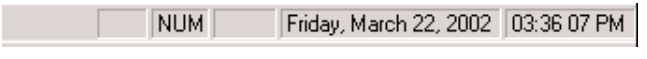
*With the **Horizontal Guidelines** switched on, the guidelines set in the **View** menu will be shown.*

Toolbar

Use this function to switch on and off the toolbar with the file and print icon, etc. at the top of the screen. You can use the toolbar to simply select for example, Open or Save, by clicking on the toolbar icon.

Status bar

Use this function to switch on and off the status bar with the date and time at the lower side of the screen.



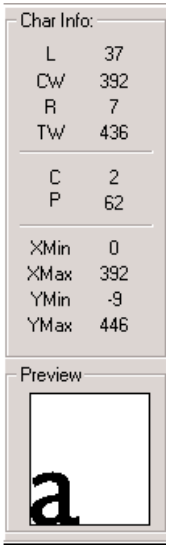
Display Function bar

Use this function to switch on and off the display functions toolbar at the top of the screen. You can use the display functions toolbar to simply hide or display, for example, marks and grid, by clicking on the toolbar icon.

Function Toolbar

Use this function to switch on and off the Function Toolbar at the right side of the screen. You can use the Function Toolbar to simply select, for example, shift or *Zoom*, by clicking on the toolbar icon .

You can also use the construction tools, such as *Circle* or *Rectangle*, etcetera by clicking on the toolbar icon.



*The functions in the Function Toolbar (top) are described in the chapter about the **Tools** menu.*

Charinfo

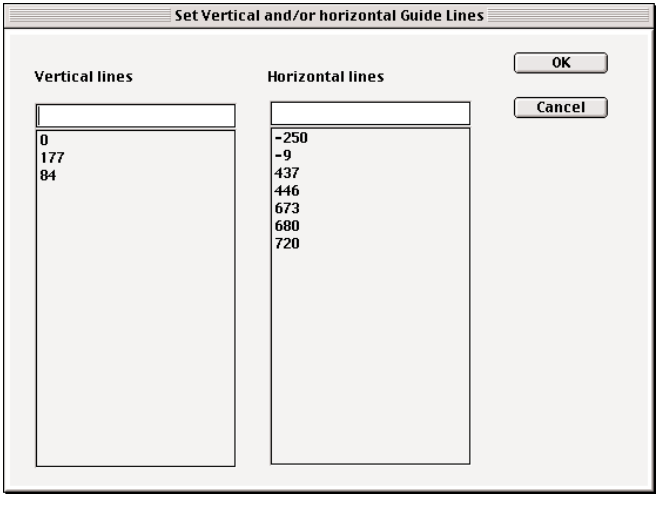
Hides or displays an information window for the characters metrics. The information will be shown on the right side of the screen together with a small filled preview of the character.

The explanation of the number is as follows:

L	(left side bearing; 1/100 mm)
CW	(character width; 1/100 mm)
R	(right side bearing; 1/100 mm)
TW	(total width; 1/100 mm)
C	(number of contours)
P	(digitizations; number of BE points)
xMin	(position of leftmost BE point)
xMax	(position of rightmost BE point)
yMin	(position of lowest BE point)
yMax	(position of uppermost BE point)

V/H Guide Lines (⌘ + ⇧Shift + L) (Ctrl + ⇧Shift + L)

You can determine vertical and horizontal guidelines in the following dialog. The values for the vertical guidelines start from the left sidebearing (LSB) of the character. After inputting a value, press the <↵Return> key.

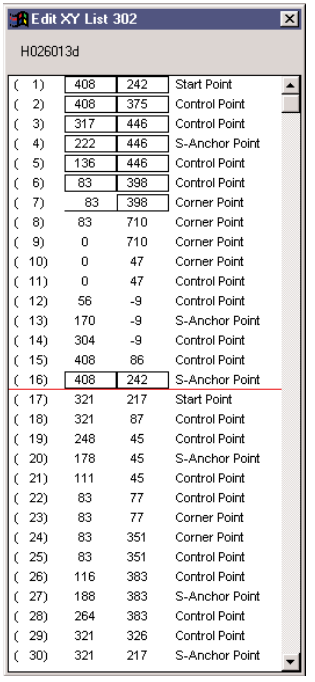


*The horizontal guide lines for ascender, x-height and descender are automatically generated based on the values in the Font Header in the **Edit** menu.*

Edit Coordinates (⌘ + E) (Ctrl + E)

Here you can edit the coordinates of all BE points in the following submenu. Selected points are marked in the display as well as in the list.

To shift the selected points numerically just double click on one selected value in the Edit Coordinates window and change it to the desired value by keyboard input. If several points are selected all points will be modified with the same amount. Undo/redo is possible via <Ctrl> + Z/Y.

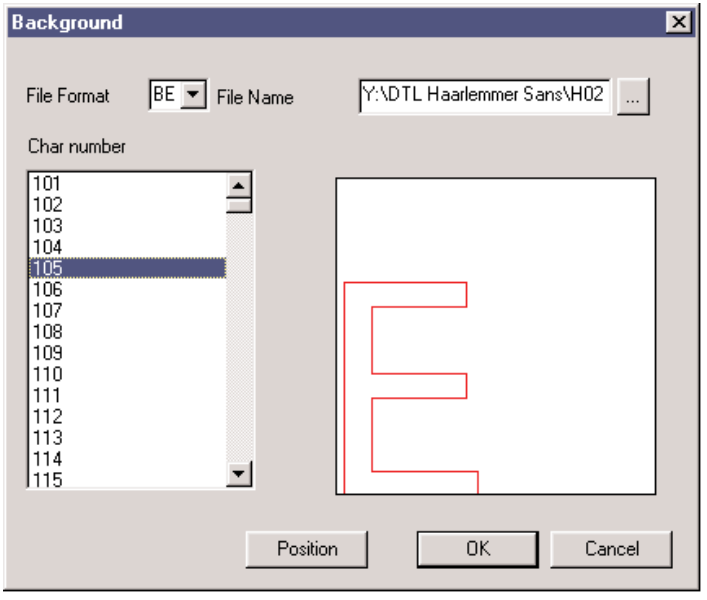


In the Edit Coordinates window the coordinates can be edited nummerically.

Select Background (⌘ + ⇧Shift + B) (Ctrl + ⇧Shift + B)

You can select another BE character from the same font or another font as a background character.

It is also possible to display a scanline character (from scanned input) as background. You can select the background from the dialog shown below.



You can select different formats (IK, II, BE and SC) for the background. It is possible to show IK and SC in the background simultaneausly.

If a background character is selected from a font database that contains more characters, typing ⌘ + → or ⌘ + ← will show not only the next or previous character in the foreground but also the next or previous character from the background font. Closing the Character Edit Window will remove the link to the background font.

Background on/off (Ctrl + B) (Alt + B)

With this function you enable or disable the background. You can also determine whether the outline font in the background is shown with or without marks or modify the position of the glyphs.

With Show background you can make the background visible.

If the Character List Window is active the **View** menu will only display the options *Toolbar*, *Status Bar* and *Font Administration*.

The *Toolbar* and *Status Bar* functions have been described in the previous pages.

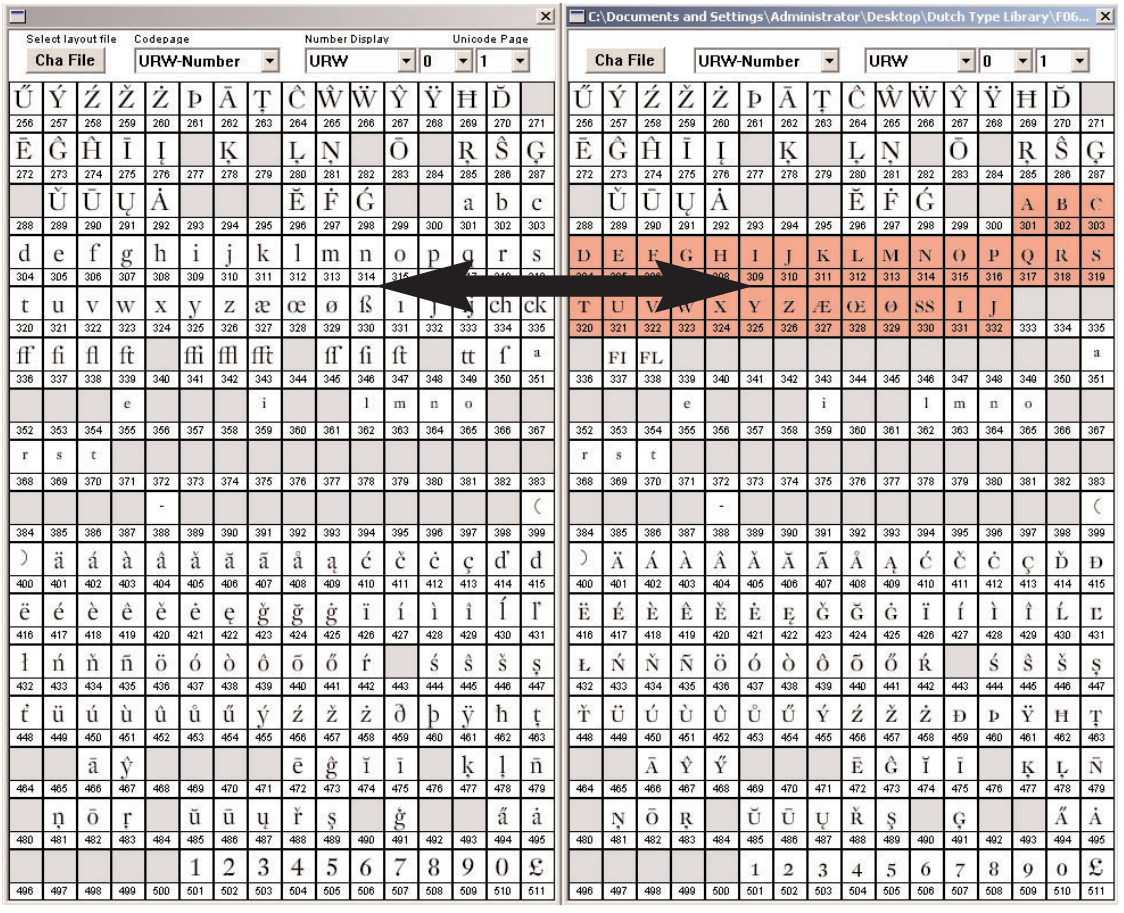
Font Administration (⌘ + U) (Ctrl + U)

This is a very powerful tool for handling and organizing the font database. Because allowed numbers are in the 16-bit range, the database can consist of more than 65.000 characters. In the Font Adminstration window the different code pages that are supported by the character set in the database can be shown beside the Unicode and the Character Numbers. It is possible to copy and paste between the codepages. Pasted characters will be placed in the database automatically under the appropriate number. Newly (re)placed characters are saved automatically. Be aware that undo is not available here!

TIP: Because changes made to the database with the Administration Tool are irreversible, it is recommended to make a back up of the font first.

The current beeditor.cha file supports the following codepages:
–Unicode
–URW-Number
–PC-West
–Mac-West
–PC -East
–Mac -East
–PC-Greek
–Mac-Greek
–PC-Turkish
–Mac-Turkish
–PC-Cyrillic
–Mac-Cyrillic
–PC-Hebrew
–Mac-Hebrew
–PC-Baltic
–Mac-Romanian
–PC-Symbol
–Mac-Symbol
–PC-Kazakh
–Mac-Kazakh

Although characters can be copied and pasted between different BE databases in the Character Edit Window, this functionality is limited to one character at the time. With the Font Administration tool it is possible to copy and exchange (large) ranges of characters between different databases. To select more than one character, hold down the <⇧Shift> key while you select single characters or several series of characters. Characters can be selected in serie by holding the mouse button down and simply dragging the mouse.



With the Font Administration tool characters can be exchanged between different font databases.

The Character Edit Window can also be opened from the Font Administration tool by double clicking on a character.

Different Character Layout Files (*.cha) can be selected. These Character Layout Files are also used by DTL DataMaster when fonts are generated. After installing DTL FontMaster four Character Layout Files are installed in the same directory as the FM modules:

–beeditor.cha

This is the default Character Layout File for DTL BezierMaster.

–TTBAS.CHA

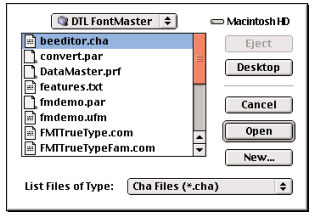
Basically you can ignore this Character Layout File here because normally it will only be used in DTL DataMaster to import fonts.

–urwotf.cha

In DTL DataMaster this Character Layout File must be selected when you want to generate OpenType. All possible OpenType features will only be generated if this .cha file is selected.

–winuni.cha

Basically you can ignore this Character Layout File here because normally it will only be used in DTL DataMaster to import fonts.



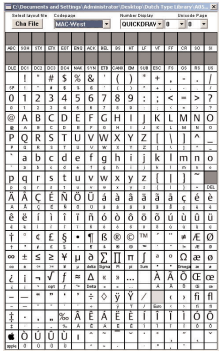
The Character Layout Files are placed in the same directory as the FM modules, libraries and the other stuff.

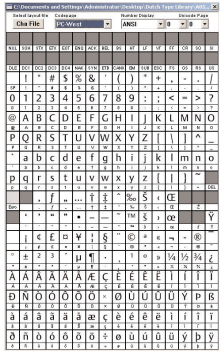
Currently, the beeditor.cha file supports 21 code pages. The Character Layout Files are fully editable and more code pages can be added by the user (see Appendix III).

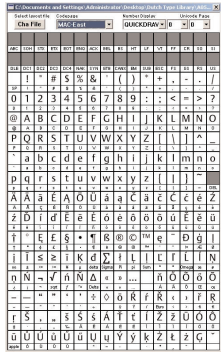
The Number Display shows five options that can be used in combination with the codepages:

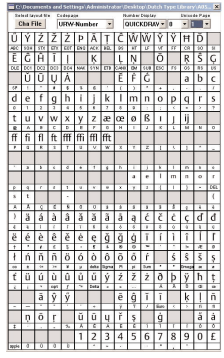
–HEX (hexadecimal notition; default for Unicode)
–URW (URW database numbers)
–ANSI (default for PC codepages)
–QUICKDRAW (default for Macintosh codepages)
–DECIMAL (decimal notition)

Although there are some default combinations of codepages and number displays, each possible combination can be made.







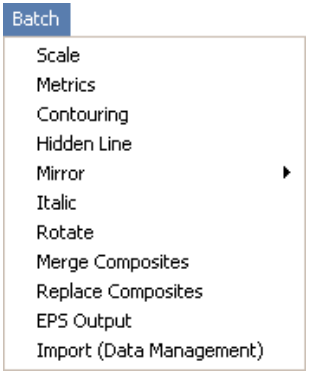


From left to right the code pages for Mac-West, PC-West, Mac-East, and URW-Number.

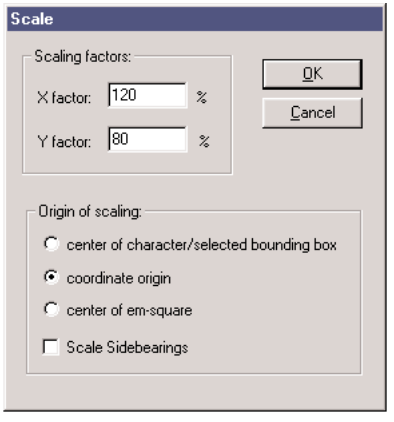
BATCH MENU

This menu is only available in case the option *Font Administration* is selected from the **View** menu. The functions in the **Batch** menu can also be found in the **Tools** and **Special** menus when the Character Edit Window is active. The big difference between the functionality in the **Batch** menu and the other two menus is that in the *Font Administration* tool several or even all characters can be selected for processing while using the same functions in the Character Edit Window only influences the selected character. If no glyph is selected the batch operation will be applied to all glyphs in the database.

Please pay attention to the fact that changes are irreversible; the operations applied to the glyphs are automatically saved on the disk.



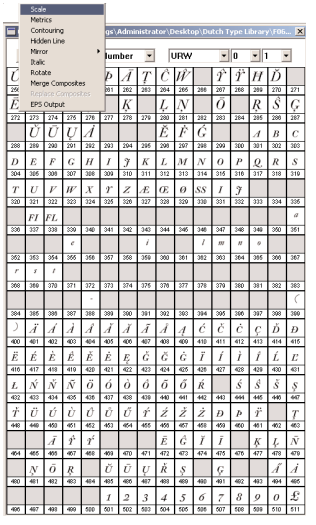
*The options in the **Batch** menu are only available in case the Font Administration tool is active.*



Scale

After having selected contours or the whole character, input the scaling factors in X and Y direction in %. You can also determine the origin of the scaling and whether or not you will scale the sidebearings simultaneously.

In case no glyph is selected the batch operation will be applied to all glyphs in the database.

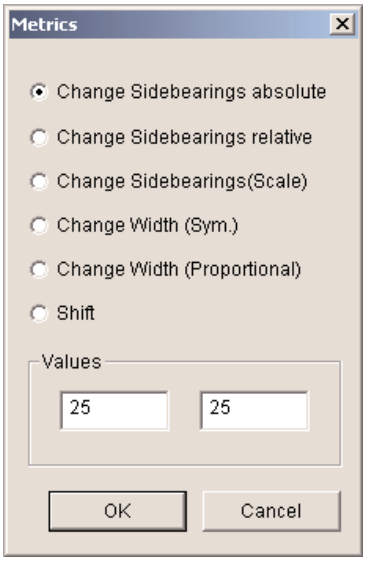


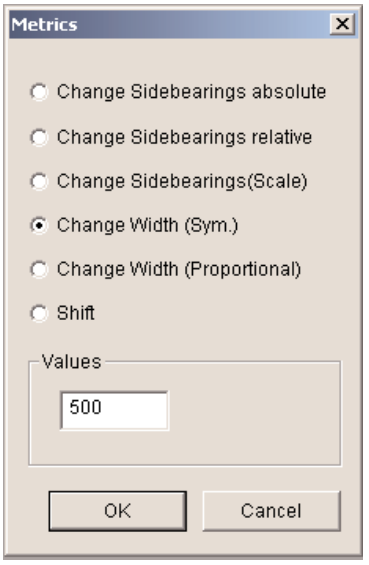
Metrics

There are several options to modify the metrics by changing the widths or sidebearings of the selected glyphs. In the Metrics dialog positiv or negative values can be entered in the input field(s) called *Values*.

– Change Sidebearings absolute

The sidebearings of all the selected glyphs will get the values specified in the two input fields for the left and right sidebearing.





For changing the sidebearing(s) the dialog will show two input fields ('Values'): one for the left and one for right sidebearing. For changing the width(s) there is, of course, only one input field available.

– Change Sidebearings relative

The sidebearings of all the selected glyphs will be modified by the values specified in the two input fields (for the left and right sidebearing).

– Change Sidebearings (Scale)

The sidebearings of all the selected glyphs will be scaled according to the values specified in the input field.

– Change Width (Sym.)

The total width of all the selected glyphs will be made equal to the value in the input field. The glyphs will be centered in the specified width, this way making the left and the right sidebearing equal (symmetrical).

– Change Width (Proportional)

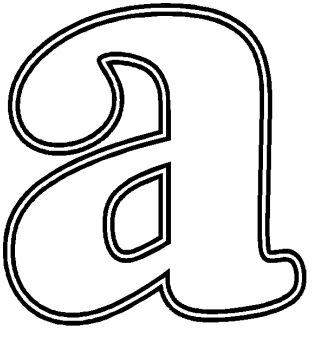
The total width of all the selected glyphs will be made equal to the value in the input field. The left and the right sidebearings will be changed proportionally according to the original relation between the sidebearings.

– Shift

The glyphs will be shifted in horizontal or vertical direction by the specified amount.

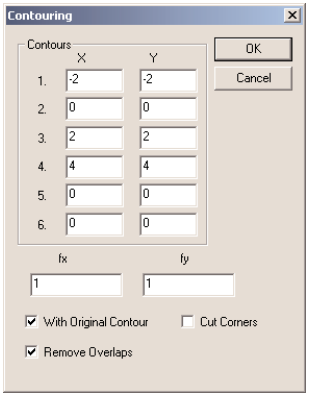
Contouring

This function is used to create additional contours automatically, so called outlined characters. It can also be used to bolden a typeface or make it



An example of contouring.

thinner. Input up to six values in millimetres into the Contouring dialog to create up to the same amount of additional contours. A positive value of for example 2 adds a contour with two millimetres distance from the original contour to the outside. A negative value works to the inside.



Up two six contours can be generated using the Contouring function from the ***Batch*** *menu.*

– Contours

Here you can enter positive or negative values for the contouring.

– fx and fy

The creation of contours can be combined with scaling. The factors specified for scaling in horizontal and vertical directions will be applied on the selected glyph(s).

– With Original Contours

In case this option is selected the original contour will be preserved. This only works if the specified values are positive.

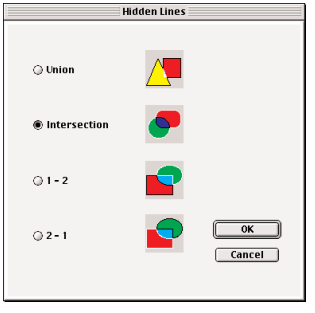
– Cut Corners

Selecting this option will preserve line thickness in the newly generated contour.

– Remove Overlaps

The functionality of this option is comparable with the *Union* function from the *Hidden Line* option and prevents overlapping contours.

The Hidden Line dialog.



Hidden Line

This function merges overlapping contours. It currently works always on the complete character.

– Union

Merges the contours.

– Intersection

Creates the intersection. Only the overlapping parts remain. The rest of the contours are deleted.

– 1-2

Deletes the second contour and the part of the first contour that was overlapped.

– 2-1

Deletes the first contour and the part of the second contour that was overlapped.

Mirror

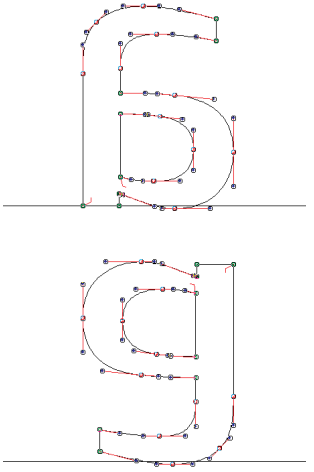
This function mirrors the selected contours or the whole character. There are two options.

Left <–> Right

This function mirrors the selected contours or the whole character horizontally around the center of the selected parts.

Top <–> Bottom

This function mirrors the selected contours or the whole character vertically around the center of the selected parts.



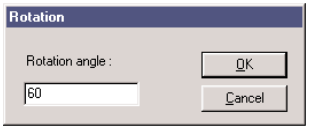
The a on top is mirrored Left <–> Right and the other Top <–> Bottom.

Italic

Use this function to oblique the character electronically. A special selection mode is not required. In this function the character mode is always used. After selecting the function a pop-up menu appears. An angle between –45 and +45 degrees is recommended. A positive angle obliques clockwise.

Rotate

The rotate function works for selected contours or the whole character. Input a Rotation angle; a positive angle rotates clockwise.



The Rotate dialog in which the angle can be specified.

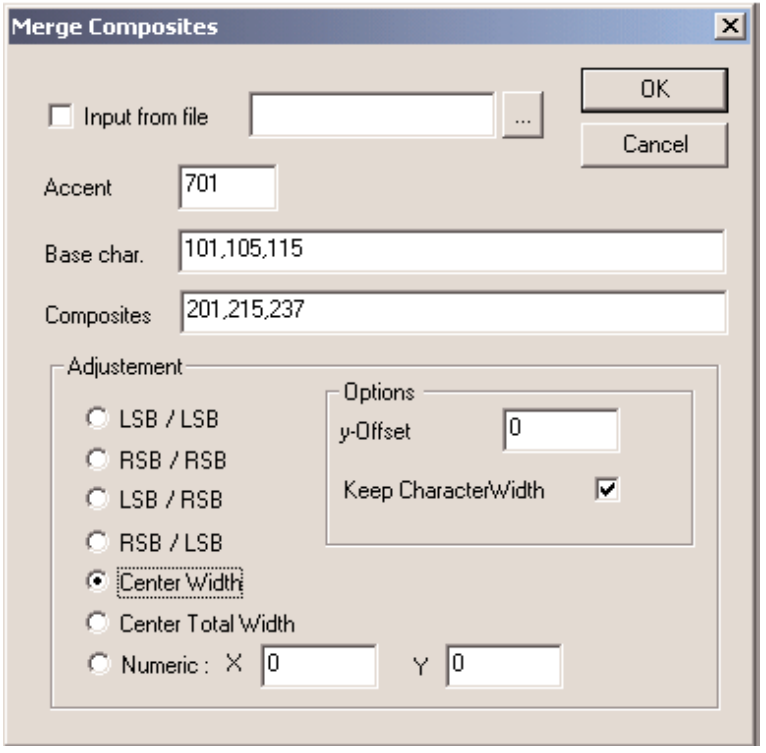
Merge Composites

This is an extremely powerful function that creates accented characters or fractions or other glyphs made out of several composites. It allows a manual parametrization or it can alternatively read and create composites from an external text file.

Selecting the *Merge Composites* function will open a dialog which shows a range of options.

– Accent

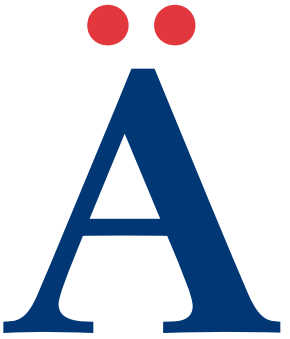
This makes together with the options *Base char.* and *Composites* manual parametrization possible. In the *Accent* input field the Character Number of the accent is specified. For instance number 701 indicates the upper case dieresis.



The Merge Composite dialog.

– Base char.

Here the Character Number(s) must be entered of the character(s) that should be combined with the specified accent. Combining the character numbers 101, 105, 115 which stand for repectively A, E and O with Character Number 701 (Dieresis) will result in Adieresis, Edieresis and Odieresis. Of course these newly generated characters have to be saved at the appropriate positions. Therefore the *Composites* must be specified.



The Adieresis was generated by combining the accent called dieresis with Character Number 701 with base character A, which has character Number 101.

– Composites

Here the Character Number(s) must be entered of the character(s) that are the result of combining the specified accent(s) with the specified base character(s). The standard positions in a BE or IK database for instance for Adieresis, Edieresis and Odieresis are the character numbers 201, 215 and 237. The order of the specified composite Character Numbers must be corresponding with the specified base characters. For instance base character 101 (A) plus dieresis *must* result in Character Number 201 (Adieresis) in case the default Character Layout Files *beeditor.cha* and *urwotf.cha* are used to generate fonts in DTL DataMaster. Details about the Character Layout Files are revealed in Appendix III. More information about the Character Numbers can be found in Appendix IX.

NOTE: The standard places in a BE or an IK database for the lowercase accents are in the range 751–769. The capital accents should be placed in the range 701–719. Normally the lowercase accents are positioned in such a way that no shifting is necessary when they are placed on top of a lowercase character. The same is the case for the capital accents. It is recommended to check first if the capital accents are available and at the right position before generating composite characters.

– *Adjustment*

For the positioning the same options are available as for the *Merge Character* function for individual glyphs from the **Function** menu.

Please note that it is important for merging accents to switch on the *Keep CharacterWidth* option, otherwise the character width of the composite will be different from the base character depending on the adjustement option.

– *Input from file*

Instead of entering all base characters, accents and composites manually the batch mode allows the input from a text file.

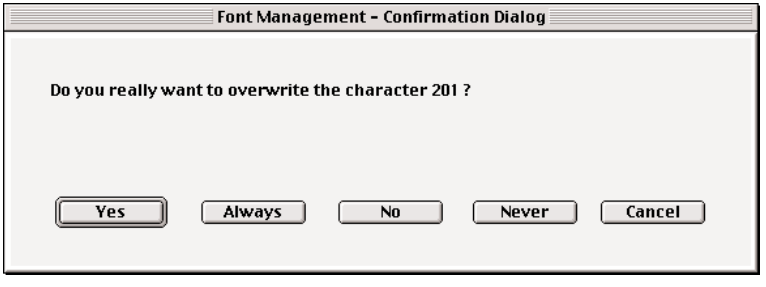
This file should have the following form:

```
// URWNum;URWComp;URWComp
201;101;701
202;101;704
```

The first number is the Character Number of the new composite glyph, the following numbers are of the glyphs which are merged, for instance respectively of the base character and the accent.

It is allowed to specify more than two components, for example to create fractions: 681;623;553;566. In combination with the option *Center Width* and *Keep CharacterWidth* this series of Character Numbers creates a nut fraction (with the Character Number 681) with the width of the first glyph (623 = hyphen) from the three glyphs 623,553 and 566.

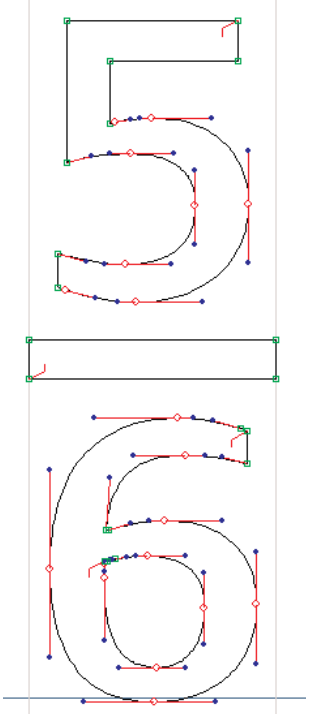
A default text file named *accents.txt* is installed in the same directory as the other DTL FontMaster files. This text file covers all characters containing accents for the Western and Eastern European and Turkish character sets for Mac OS and Windows.



In case the **OK** button has been pushed, the *Font Management-Confirmation Dialog* pops up in case character numbers already exist in the database. *Always* means that all existing characters with the same numbers will be replaced. Please note that overwriting characters is irreversible.

```
// URWNum;URWComp;URWComp
201;101;701
202;101;704
203;101;705
204;101;706
206;101;708
207;101;709
208;101;703
209;101;713
210;103;711
211;103;704
212;103;707
214;104;707
```

The default text file for making composites, accents.txt, is installed in the DTL FontMaster directory.



A nut fraction that was automatically generated using the Merge Composites function from the ***Batch*** *menu.*

Replace Composites

With this function the base characters used for the composites can be replaced by their originals. This is very useful when changes have been made to the base characters after the generation of the composites with the *Merge Composites* function. Please note that the accents are not affected and these have to be replaced manually in case of changes made to the originals.

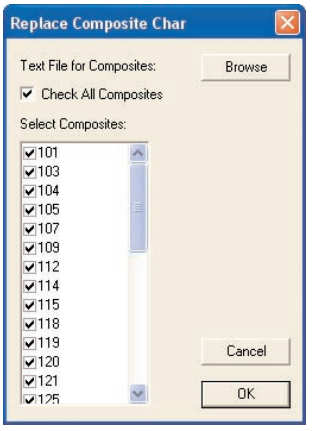
Selecting the *Replace Composites* function will open a simple dialog which shows a small range of options.

– Text File for Composites: Browse

Here a text file that contains composite information, like for instance the default *accents.txt*, has to be selected. The second entries (after the actual Character Numbers of the composites) in the text file will show up in the *Select Composites:* part of the dialog.

– Check All Composites

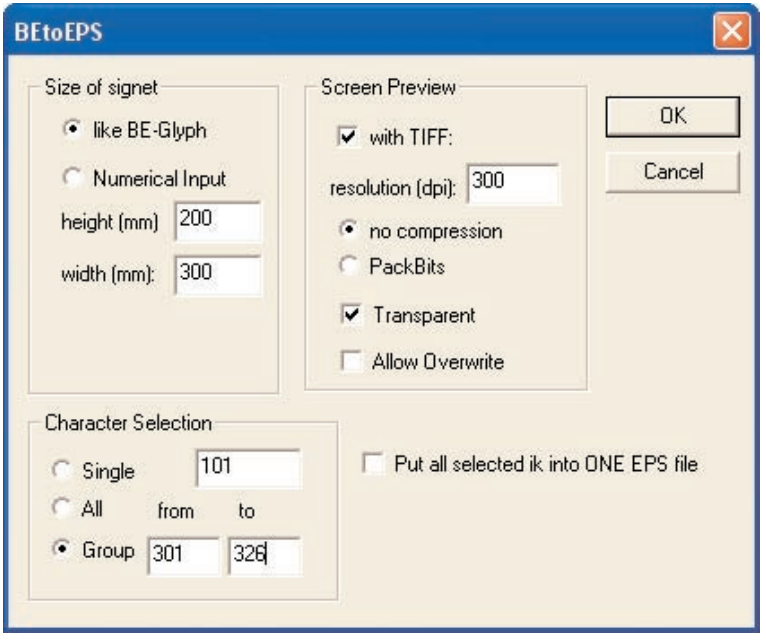
This option makes it possible to replace all base characters listed in the selected text file or, when not activated, to select individual Character Numbers in the *Select Composites:* part of the dialog.



After a text file that contains the composite information has been selected, the Character Numbers of the base characters show up in the dialog.

EPS Output

Glyphs can be exported as EPS data individually or as group. Selecting this batch function will open the *BE to EPS* dialog containing a range of options.



Glyphs can be exported as EPS data using several options.

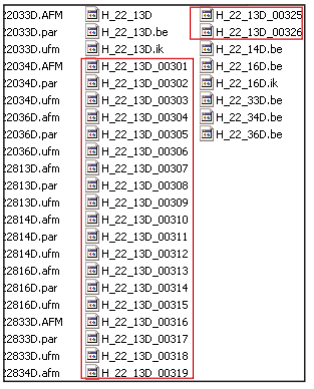
– *Size of signet*

The glyph(s) can be exported with unchanged size by selecting the option *like BE-Glyph*. This preserves the original relation to the other glyphs in the font database, which makes re-importing (for instance after editing in Adobe Illustrator®) possible without any scaling. With the option *Numerical Input* the bounding box can be scaled to any size measured in millimeters.

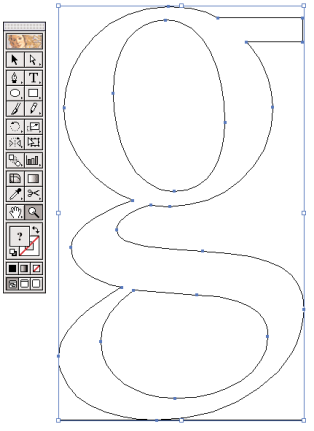
– *Screen Preview*

The EPS files can be provided with a TIFF for screen previewing. As an option the required resolution can be entered. The default reolution is 300 dpi. The TIFF files can be compressed with Apple's *PackBits* format or leaved uncompressed, which is the default setting.

Further options are *Transparant* and *Allow Overwrite*, which let the program overwrite EPS files with the same name. The name of the EPS is taken from the name of the database plus the Character Number. Exporting the glyph with Character Number 302 of the H022013D named database will result in H022013D_00302.EPS. The EPS files are automatically stored in the directory that contains the used font database.



The exported EPS files are stored in the directory that contains the used font database.

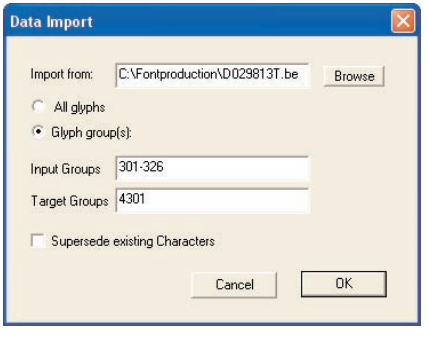


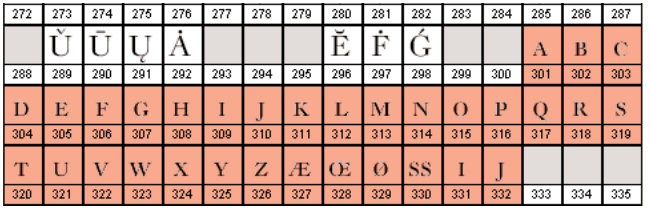
The size of the exported EPS file is based on the bounding box of the glyph.

– *Character Selection*

Glyphs can be exported individually as EPS files but also as group. With the option *Single* a Character Number can be entered. The option *All* will export all the glyphs in the font as EPS files. It is also possible to export ranges using the input fields of the *Group* option.

If multiple glyphs are exported by default the prgram will generate a number of single EPS files. In case one large EPS file that contains all glyphs is required, the function *Put all selected BE into ONE EPS file* should be activated.





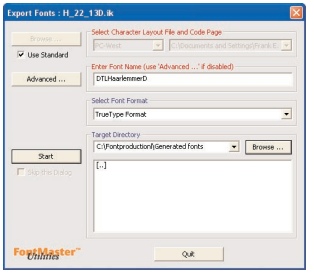
The Import (Data Management) function makes the exchange of data between different files very easy.

Import (Data Management)

With this very powerful batch function glyph data can be imported in the currently open BE database from any other BE database. It is not necesarry

to actually open the other database(s) in DTL BezierMaster. This function makes it for instance possible to build large databases from ones that contain only single code pages, which is very useful especially with the OpenType production in mind.

Using the default Character Layout File *beeditor.cha* when importing for instance PostScript Type 1 fonts in DTL DataMaster will place the lower case characters in the Character Number range 301–326. The same will happen when a Small Caps font is imported because of the used PostScript names for the characters. However, the 'real' names of the small caps are not a,b,c, etc. but Asmall, Bsmall, Csmall, etc. The corresponding Character Numbers in the *beeditor.cha* are 4301, 4302 and 4303. By simply entering the range 301–326 in the *Input Groups* input field and entering 4301 in the input field of *Target Groups*, the small caps will placed in the correct positions in the database, which correspondents with the appropriate PostScript names and Unicode numbers.



The default setting for the Character Layout File in DTL DataMaster will let the program use the beeditor.cha file when PostScript Type 1 and TrueType fonts are converted to the BE format. For OpenType fonts the urwotf.cha file is used by default.

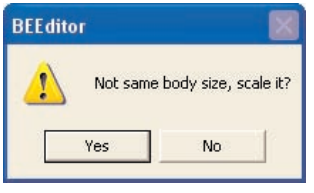
– Import from:

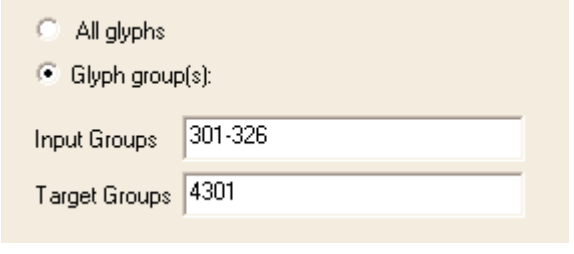
Here the BE database has to be entered from which glyphs have to be imported in the currently active font in the Font Administration tool. You can browse to select the database.

Activating the *All glyphs* option will import all the characters from the selected database. If the *Glyph group(s):* has been activated subsequently the range(s) of the *Input Groups* and the *Target Groups* have to be defined. In the input filed of *Input Groups* the first and last Character Number has to be given divided by a hyphen. For the *Target Groups* only the first Character Number has to be entered. Comma's are used between multiple groups. For example: the groups 101-110, 301-310 entered in the *Input Groups* section can be placed at the *Target Groups* 4101,4301. There is basically no limit to the number of groups defined.

If the *Supersede existing Characters* check box has been activated, existing glyphs with the same Character Numbers as entered in the *Target Groups* input field, will be overwritten. Please pay attention to the fact that these changes are irreverible!

In case the bodysize of the two databases are different, you will be asked whether to scale or not the glyphs to the bodysize of the target database.

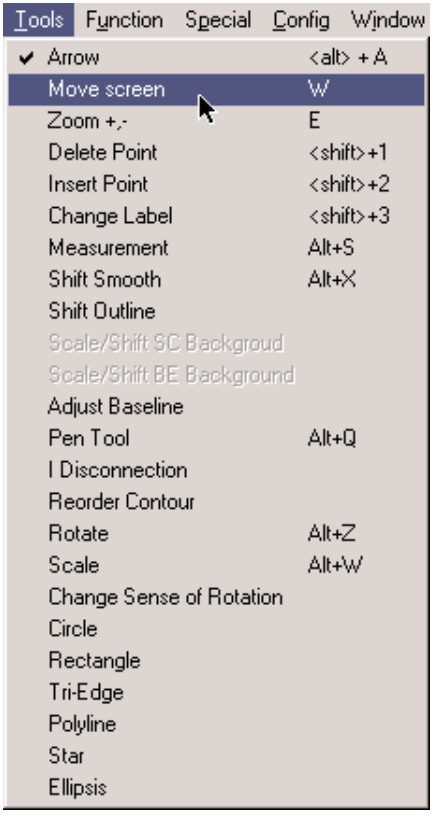




Glyphs can be imported in single or multiple groups.

TOOLS MENU

The functions in the **Tools** menu can be chosen either from the pulldown menu or from the *Function Toolbar*. Moving the mouse slowly across the toolbar automatically displays an explanation of the icon. Some functions can also be chosen via keyboard shortcuts. The abbreviations are shown in the pulldown menu on the right side, for example <Alt> + A keys to select the arrow (pointer) tool.

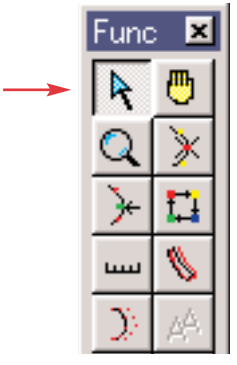


TIP: Several of the functions from the Tools menu, that work in the Character Edit Window only for the active character, can be used for a (large) range of characters using the Batch menu in combination with the Font Administration tool.

Arrow (Space) (⌥Alt + A) (Alt + A)

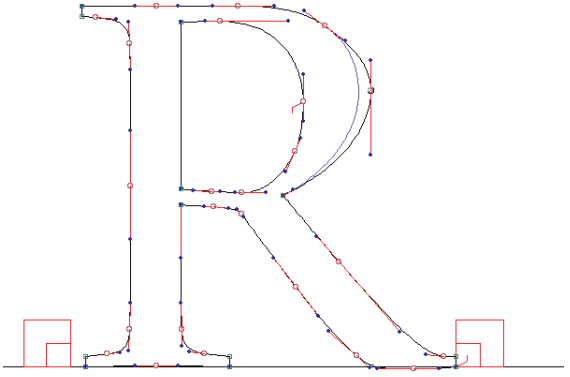
The arrow or pointer tool is the standard tool to select one or more points or contours as explained in the selection rules:

– Clicking near a point selects this point.
– <⇧Shift>+ mouse click adds a point to the selection.
– A mouse click far away from any point deselects everything.
– A selection with a rectangle can be made by holding down the mouse button and dragging it across the desired area.
– A contour can be selected by double clicking on it.
– A glyph can be selected by double clicking inside it.



The selected objects can be shifted easily with the arrow tool. Move the arrow near one of the selected objects, press the mouse button, hold it down and move it to the desired position. The objects will be shifted to that position.

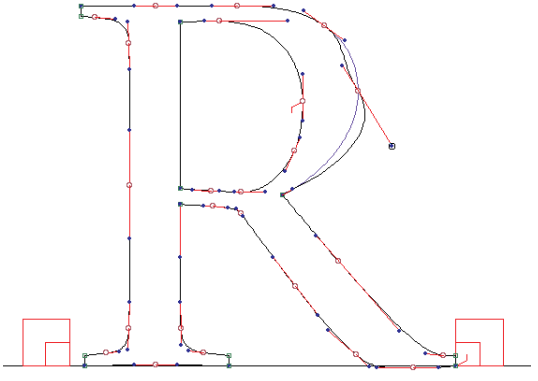
The behaviour of the different points under the shift-action is different.
–An anchor point and a smooth anchor point will be shifted without influencing other points.

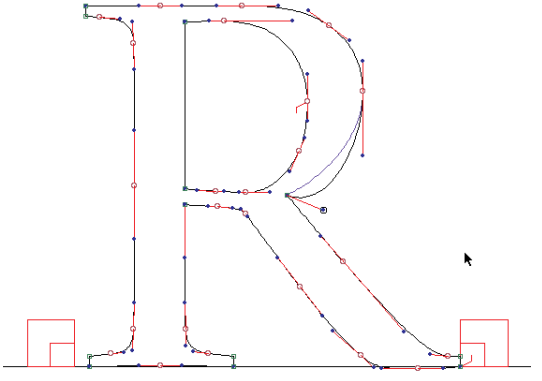


Points can be shifted with the ← and → keys also. The Step Shift units can be defined in the ***Config*** *menu.*

–A control point will be shifted as a single point if it does not connect to a smooth anchor point. If it is connected to a smooth anchor point it will influence the other control point too, in order to preserve tangent continuity.

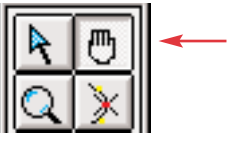
If a control point is connected to a smooth anchor point, it will influence the other control point too (left). Otherwise it will be shifted as a single point (right).

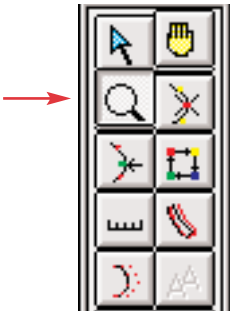




Move Screen (w) (w)

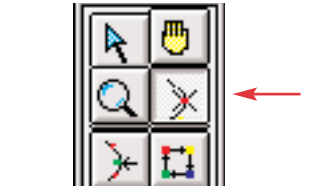
This function allows you to shift the visible display of the currently edited character within the window. It is equivalent to scrolling using the standard scrollbars, but more conveniently.





Zoom +,– (E) (E)

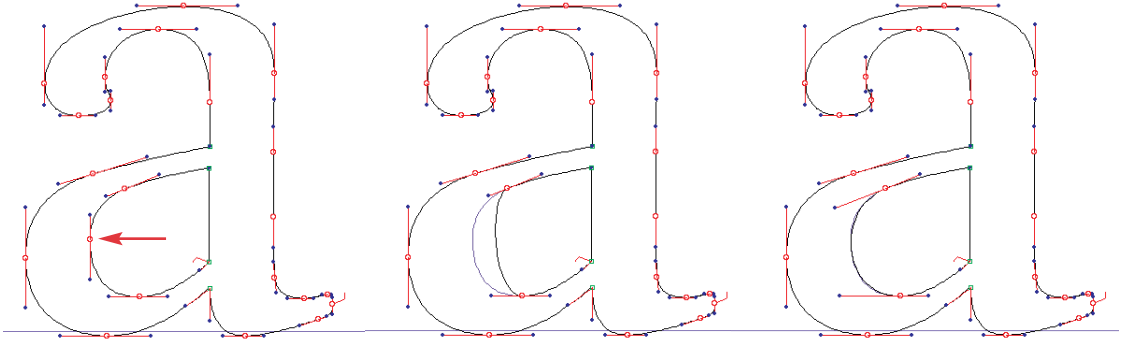
Selecting this function changes the cursor to a looking-glass symbol. You can use it to enlarge or reduce the display.
Zoom (+) Click the mouse button enlarges the display in small steps.
Zoom (-) <⇧Shift>-mouse click reduces the display in small steps.
Holding down the mouse button and dragging the mouse opens a rectangle. If you enlarge the display this rectangle will be the viewable area if the mouse button is released again. If you hold the <⇧Shift> key while dragging the mouse, the currently visible area will be displayed in the rectangle generated by the mouse. **Reset (⌘ + R)** will restore the default image size. The zoomfactor can be set in the **Config** menu.



Delete Point (⇧Shift + 1) (⇧Shift + 1)

With this function, if you click on an anchor point, it will be deleted immediately. If the anchor point delimites two Bezier curve sections the neighbouring control points will be deleted as well. If you click on a control point both control points of a Bezier section will be deleted.

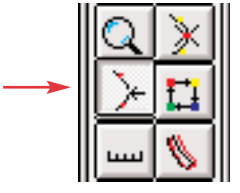
<⇧Shift>-mouse click will remove the anchor point but the shape of the contour will approximate the shape of the original as much as possible.



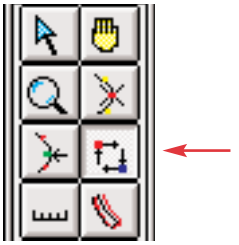
At left the original a with the point that has to be removed indicated by the arrow. In the centered a the point is removed only. The a at the right has the point removed while the <Shift> key was held and the original shape is approximated.

Insert Point (⇧Shift + 2) (⇧Shift + 2)

Click the mouse near the contour to insert a point. If the nearest part of the contour is a straight line a new anchor point will be inserted at the cursor position. If the nearest part of the contour is a Bezier curve this curve will be divided into two Bezier sections and the new anchor point will be inserted on the contour, not at the mouse position. The newly inserted point can then be shifted to the desired position.



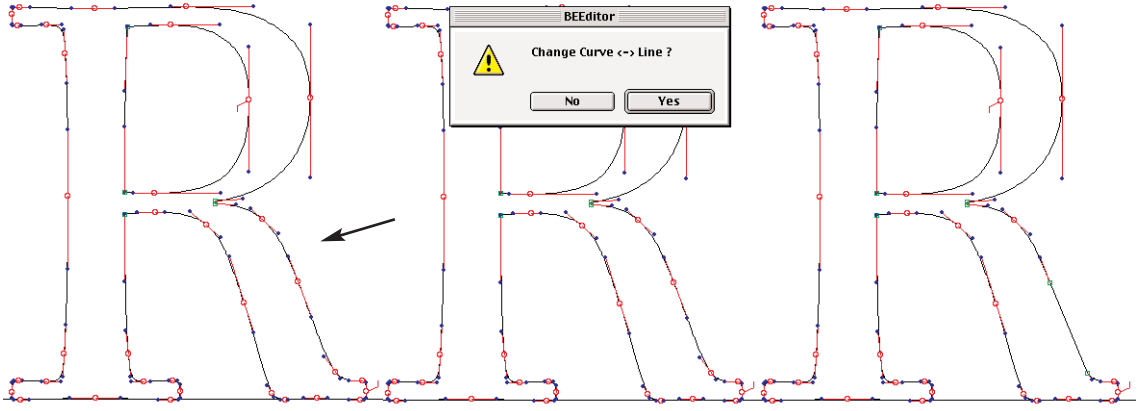
Change Label (⇧Shift + 3) (⇧Shift + 3)



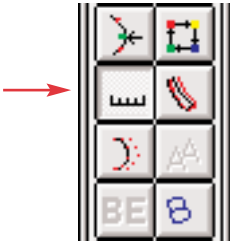
This function can be used to change a straight line into a bezier curve or vice versa.
– Click near a straight line: Two control points will be inserted
– Click near a curve: The two control points will be deleted.
This function can also be used to toggle between anchor points and smooth anchor points. If you use <Ctrl> + mouse click the program will change the label from Anchor to Smooth Anchor point and vice versa. The change in the outline is not reversible. If you change an Anchor point to a Smooth Anchor point the program will modify the tangent directions at this point and hence modify the position of the neighbouring control points.

Bezier Control Points can be simply deleted by selecting the BCP and by pressing the <←Backspace> key.

In the tail of the R the bezier curve is converted into a straight line.



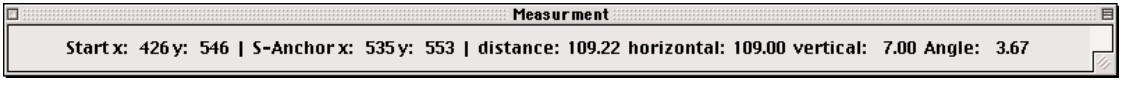
Measurement (Ctrl + s) (Alt + s)

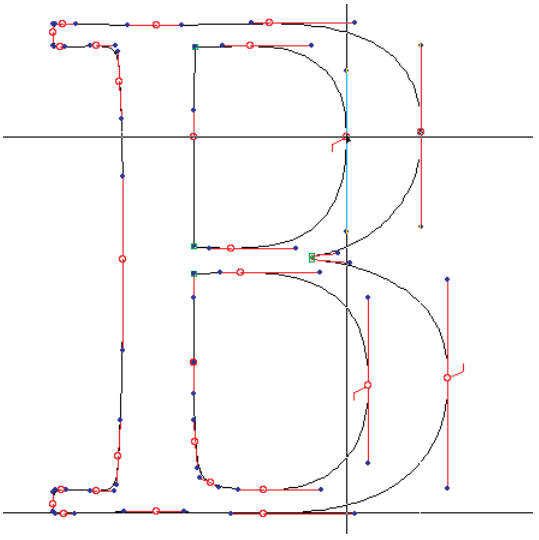


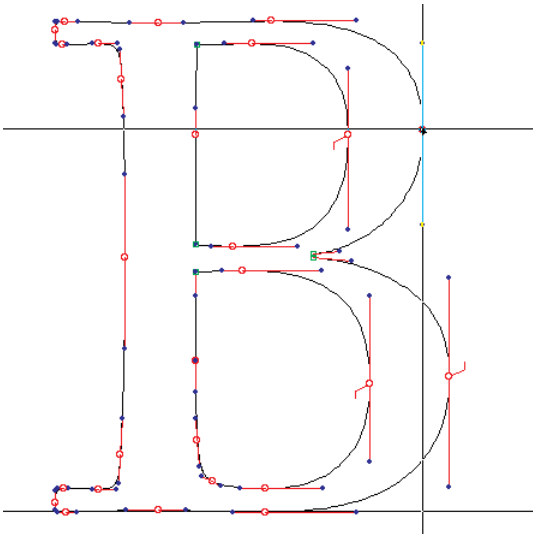
This function can measure the distance between two outline points by simply clicking on them with the mouse. The result will be shown in a small popup window on the screen. You can make the measurement window disappear by selecting another function like *Arrow*.

The measurement window displays the coordinates, the nature (start, anchor, corner, etc.). The same information is displayed for the last point which was clicked with the mouse (denoted as 'Another Point').
You will also see the distance of these two points in X and Y, the complete euclidian distance and the angle between these points.

If you click far away from any outline point the screen display position will be displayed and no information about point number etcetera is shown.



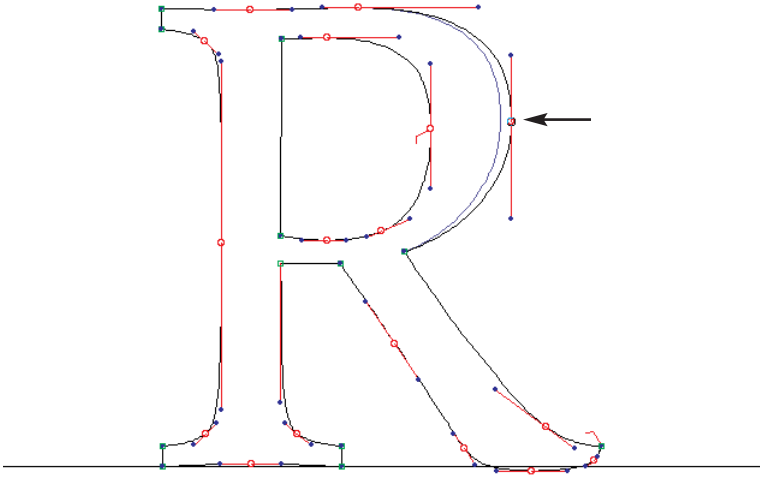


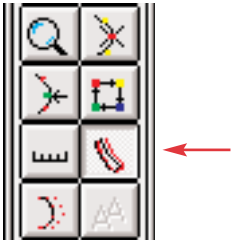


Start x: 426 y: 546 | S-Anchor x: 535 y: 553

For measuring the distance between two points, select a point with a mouseclick and repeat this with another point.

If the *Measurement* function is selected the Crosshair Cursor is automatically activated.



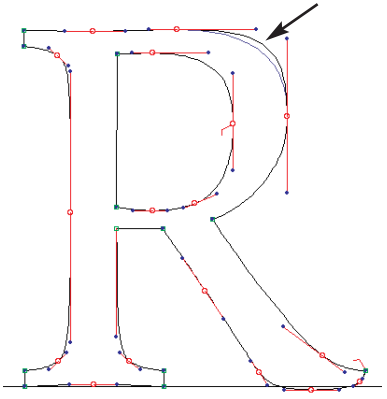


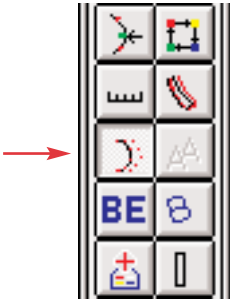
Shift Smooth (Ctrl + x) (Alt + x)

This function can be used to move points. The point will be shifted such that tangent continuity is preserved. This means that the adjacent control points are shifted too.

Shift Outline

Using this function you can shift parts of the Bezier curve by clicking onto the contour directly instead of selecting control points. *Shift Outline* is a very helpful tool for redesigning the shape of curves.



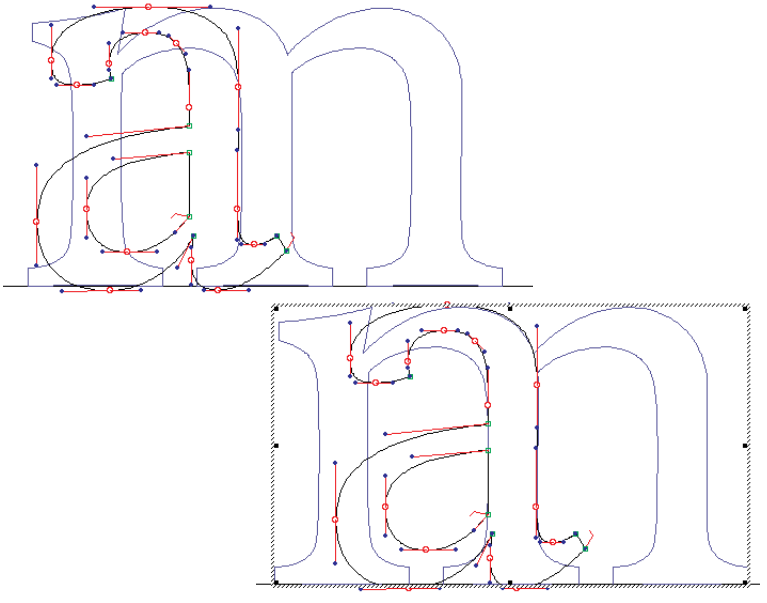


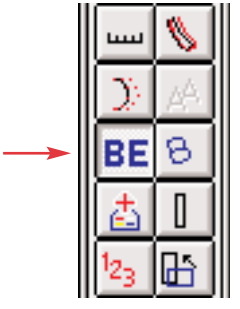
Scale or Shift SC Background

Using this function you can scroll the SC background to a desired position. After selecting this function, a scalable rectangle will appear which you can move on the screen to the desired position. Currently not implemented.

Scale or Shift BE Background

Using this function you can scroll and scale the BE background to a desired position and size. You have to set the BE background first in the View menu.

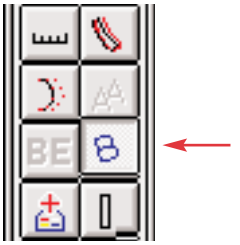




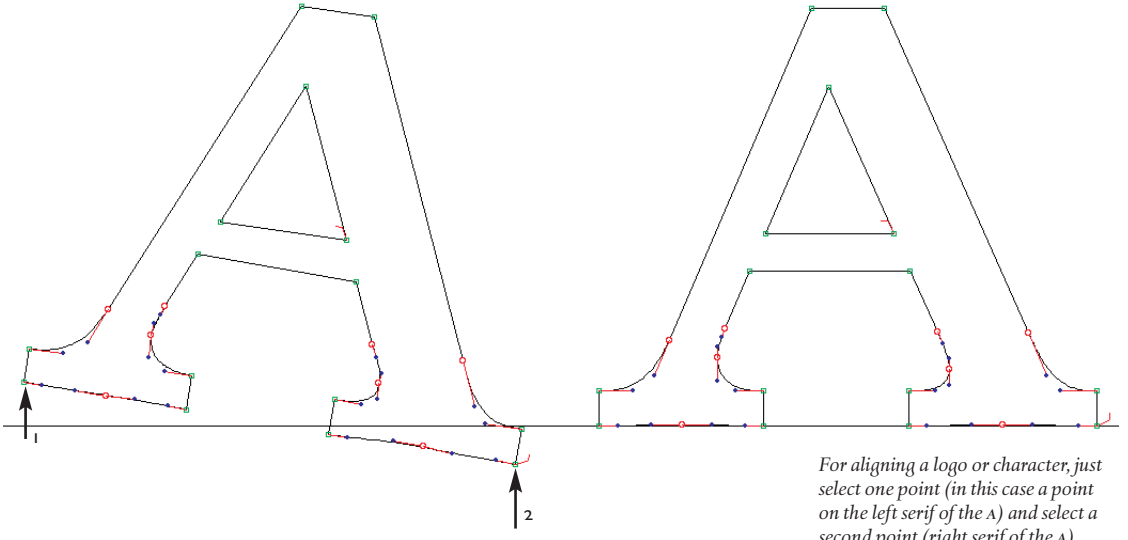
Moving the mouse into the rectangle that marks the BE background will change the cursor into a four-arrow shape. With this cursor activated, the background can be moved. Selecting one of the eight points that mark the background rectangle will change the cursor into a two arrow variation. With this cursor activated, it is possible to resize the background.

Changes made in the background are irreversable.

Adjust Baseline



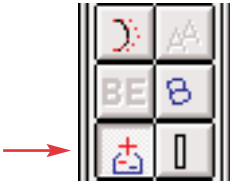
This function is designed to align logos or letters which are automatically converted from scanner data into outline data. It is not possible to align a logo 100 % vertically or horizontally by hand on a scanner. Use this function to align a logo to two outline points. The function will rotate the character so that the two selected points are either horizontal or vertical after the rotation, depending on the angle of the original points. After selecting this function, click on the first anchor or control point and then on the second one.

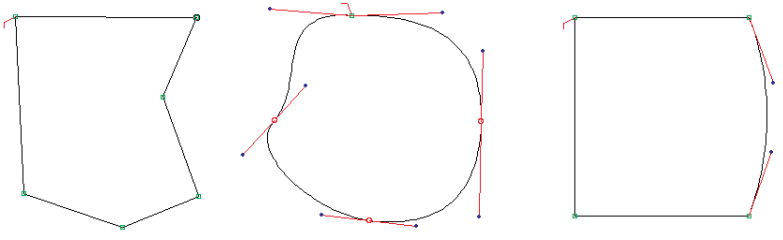


For aligning a logo or character, just select one point (in this case a point on the left serif of the A) and select a second point (right serif of the A).

Pen Tool (Ctrl + Q) (Alt + Q)

With this function you can draw contours by directly placing points with the mouse in the Character Edit Window. By just clicking and immediately releasing the mouse button, only anchor points will be placed. By clicking and moving without releasing the mouse button, smooth anchor points will be placed. While holding down the <⇧Shift> key and clicking the mouse, the points will be connected by straight lines in vertical or horizontal direction with exception of the line that closes the contour. This function makes a perfect combination with the *Change Label* function.

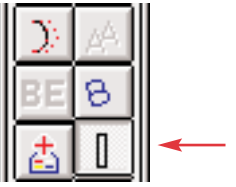


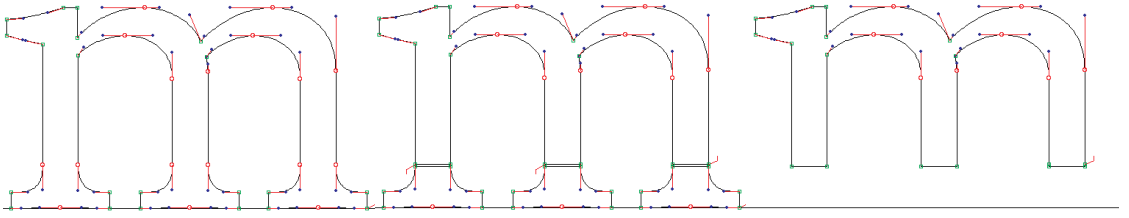


The contour at the left has only anchor points. The centered shape consists of smooth anchor points because the mouse was moved without releasing the mouse button. The shape at the right was made by clicking the mouse while holding the <⇧shift> key. The curved part was made by first using the Change Label function and moving the BCP's afterwards.

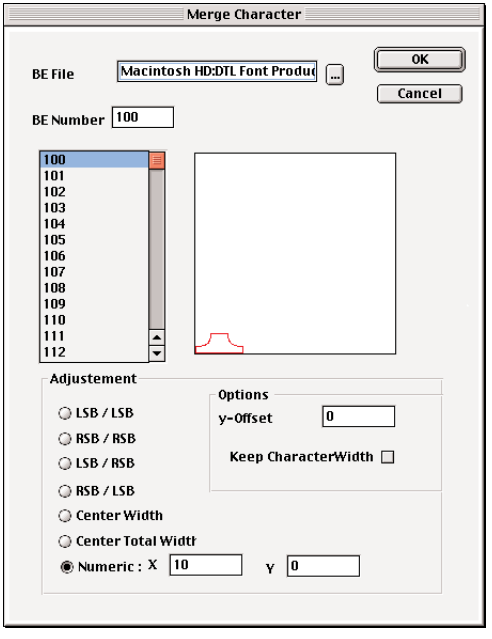
1 - Disconnection

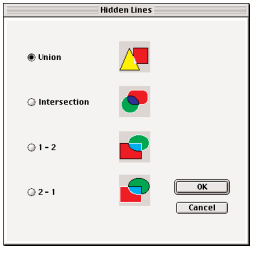
This is a powerfull function to cut a single contour is into two contours with an overlap. Especially in combination with the *Hidden Line* function, the *1-Disconnection* function can be of great value for font production. For instance, a database of different serifs can be built and connected to the stems at a later stage of the production.

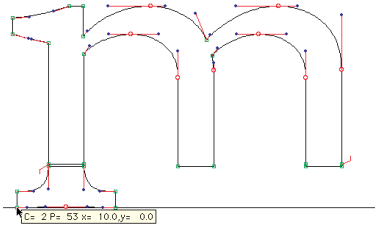




Disconnecting is very easy; just select a point with the mouse and consequently select another point. Two closed contours will be generated automatically. In the illustrations this action is repeated until all three serifs could be removed.





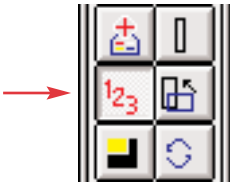


It is possible to build a database of serifs that for instance can be connected numerically to the stems using the Merge Character and Hidden Lines functions.

The selection of the location of the cut can be done by selecting two BE points (the function is executed immediately after having clicked onto the second BE point) or by dragging the mouse by holding the left mouse button down over the contour part which has to be separated. The function is executed immediately after having released the mouse button.

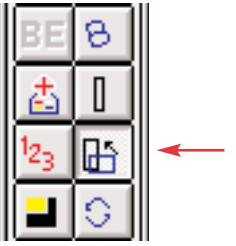
The overlap is set in the *Disconector Overlap (units)* option in the **Config** menu.

Reorder Contour

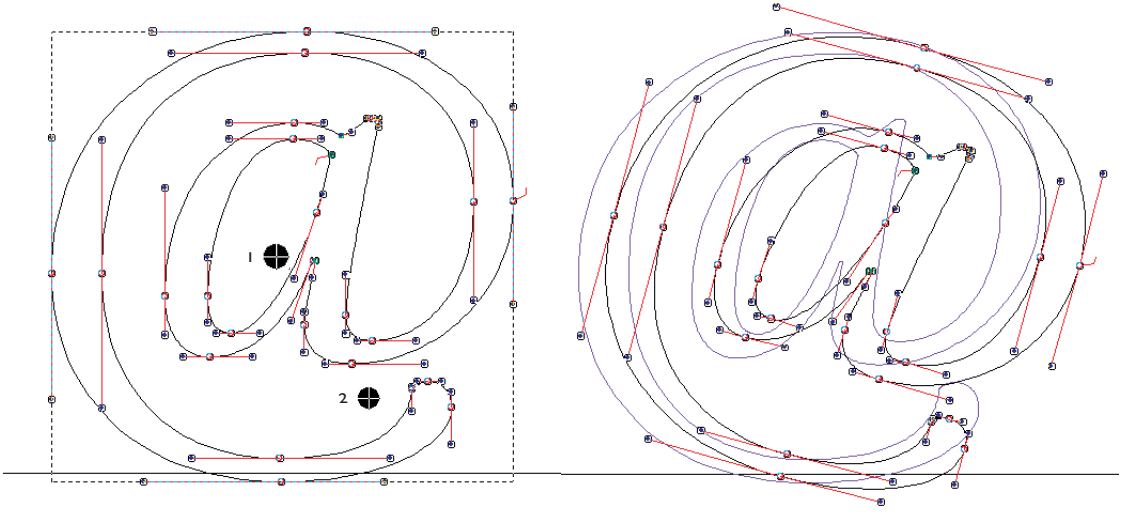


Using this function, you can reorder the contours manually. Just double click onto the contour which you want to set as the first one, then double click onto the second contour, which will become contour number 2, and so on.

Rotate (Ctrl + z) (Alt + z)

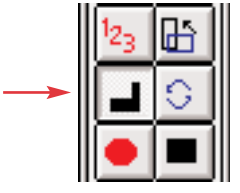


Using this function, you can rotate selected contours or the complete character around a defined center of rotation. Set the center of rotation with the mouse. The default setting is the center of the bounding box but you can place the center of rotation practically everywhere. Changing functions in the Function Toolbar wil restore the default position. To rotate the character click on the edges of the rectangle and move the mouse.



The center of rotation is by default the center of the bounding box (1) but can be placed anywhere else (2). The rotation at the right was made with the second center.

Scaling (Ctrl + w) (Alt + w)



With this function you change the size of characters, contours or contour groups. It is also possible to select individual points. In general scaling can be done either using the mouse or a numerical input via the keyboard.

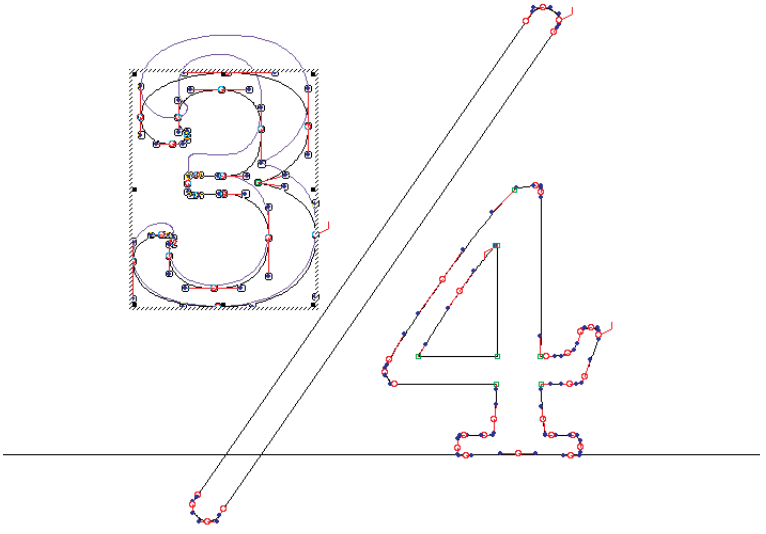
The selection of this function generates a rectangle around the selected points. You can then either scale or shift this rectangle depending on the way you drag the mouse. Selecting certain magic points of this rectangle enables different funtions:

– Diagonally to the upper left side

– Vertically to the upper side

– Diagonally to the upper right side
– Horizontally to the left side
– Central equal to all sides
– Horizontally to the right side
– Diagonally to the lower left side
– Vertically to the lower side
– Diagonally to the lower right side

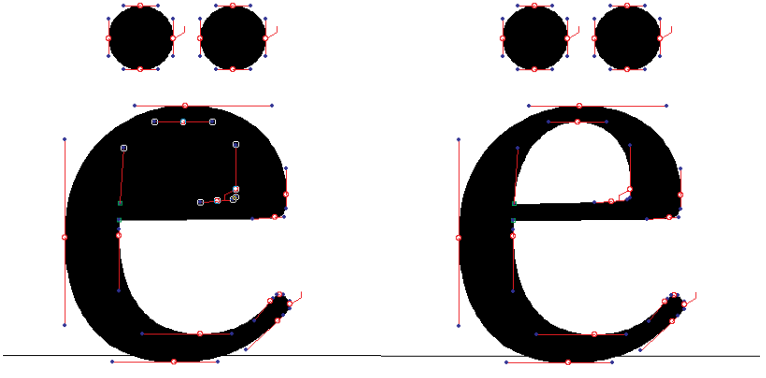
After the direction and factor of the scaling have been selected, execute the function finally by clicking again.

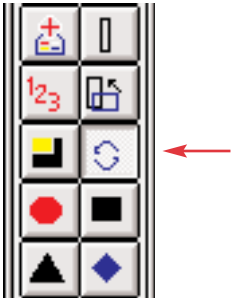


After selecting the points, objects can be scaled in several directions.

Change Sense of Rotation

This function allows you to change the path direction of a contour. Just click on an outline point of a contour to change its sense of rotation. Changing the sense of rotation affects the fill of inner and outer contours.

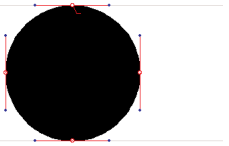




If the direction of the inner contour is not properly set, it will be filled also (left). After changing the sense of rotation this problem will be solved

Circle

This function allows you to create a circle. Select the function, click at a position you wish and drag the mouse. The circle will be generated on the fly and can be changed as long as you hold the mouse button pressed.

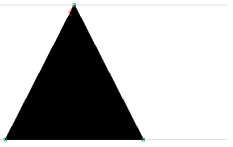


Rectangle

This function allows you to construct a rectangle. It will be generated on the fly and can be changed as long as you hold down the mouse button.

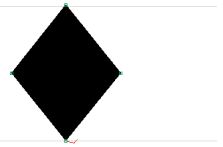
Tri-Edge

This function allows you to construct a triangle. Select the function, click at a position you wish and drag the mouse. The triangle will be generated on the fly and can be changed as long as you hold down the mouse button.



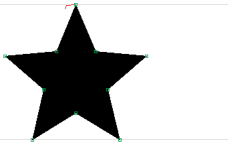
Polyline

This function allows you to construct a polygon. The number of corners for a Polygon can be set in the *Settings* function of the **Config** menu. Select the function, click at a position you wish and drag the mouse. The polygon will be generated on the fly and can be changed as long as you hold down the mouse button.



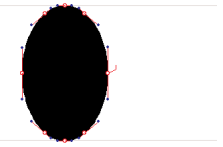
Star

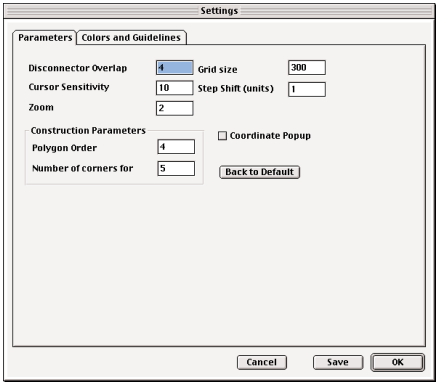
This function allows you to construct a star. The number of corners for a star can be set in the *Settings* function of the **Config** menu. Select the function, click at a position you wish and drag the mouse. The star will be generated on the fly and can be changed as long as you hold down the mouse button.



Ellipsis

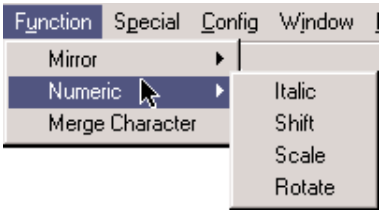
This function allows you to create an ellipsis. Select the function, click at a position you wish and drag the mouse. The ellipsis will be generated on the fly and can be changed as long as you press the mouse button.





The number of corners of the Polygon and Star functions can be set in the Settings function of the ***Config*** *menu.*

FUNCTION MENU

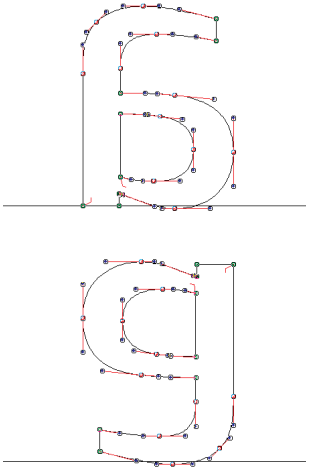


Mirror Left <–> Right

This function mirrors the selected contours or the whole character horizontally around the center of the selected parts.

Mirror Top <–> Bottom

This function mirrors the selected contours or the whole character vertically around the center of the selected parts.



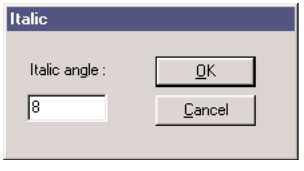
The a on top is mirrored Left <–> Right and the other Top <–> Bottom.

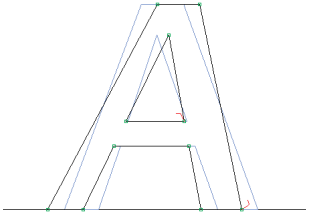
Numeric

Several functions can also be used with numerical input: *Italic*, *Shift*, *Scale*, *Rotate*. Select the contours to be modified, choose the numeric function, for example *Scale*, and then fill in the numeric values (your desired parameters) into the pop-up window, which will appear on the screen.

– *Italic*

Use this function to slant the character electronically. A special selection mode is not required. In this funtion the character mode is always used. After selecting the function a pop-up menu appears. An angle between –45 and +45 degrees is recommended. A positive angle slants to the right.

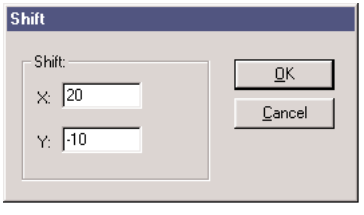


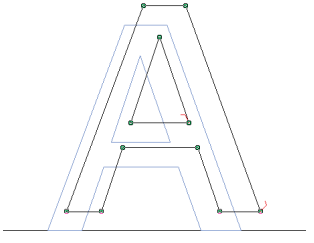


The smallcaps a in this example was slanted nine degrees to the right.

– *Shift*

This function works on all levels, for selected points, contours or the complete character.

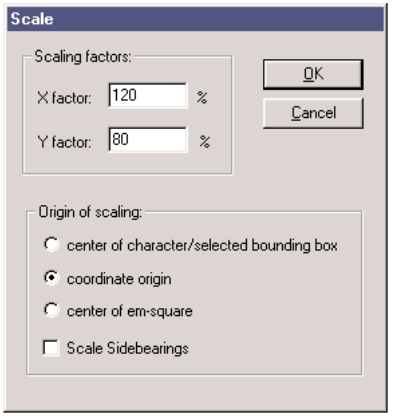


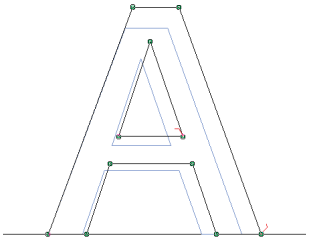


The smallcaps a in this example was shifted 50 units over the X- and Y-axes.

– *Scale*

After having selected contours or the whole character, input the scaling factors in X and Y direction in %. You can also determine the origin of the scaling and whether or not you will scale the sidebearings simultaneously.

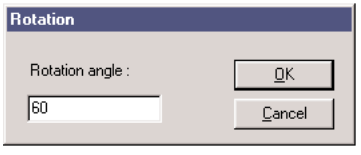


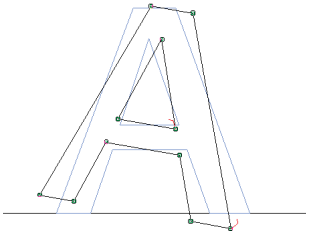


The smallcaps a in this example was scaled 110 % for the X and Y factors with the option coordinate origin selected.

– *Rotate*

The rotate function works for selected contours or the whole character. Input a value in *Rotation angle:* a positive angle rotates clockwise.





The smallcaps a in this example was rotated ten degrees clockwise.

Merge Character

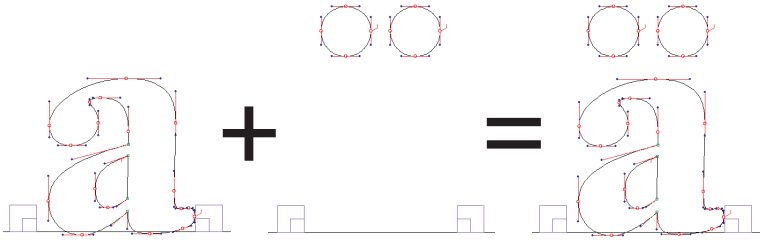
This function allows you to merge another character with the currently edited one. To do so you select this menu option. It allows to select a typeface, from which you can copy a character into your current character. You must select the character number in this typeface and afterwards fix the position for merging.

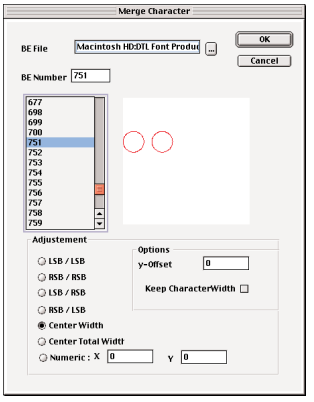
The dialog box allows different options for adjustment, which might be used for different purposes like merging accents or creating fractions or other composite characters. The options are:

– Left side bearing (LSB) / left side bearing (LSB)
– Right side bearing (RSB) / right side bearing (RSB)
– Left side bearing (LSB) / right side bearing (RSB)
– Right side bearing (RSB) / left side bearing (LSB)

– Center Width (center in the Bounding Box)
– Center Total Width (center between the sidebearings)
– Numeric: X, Y

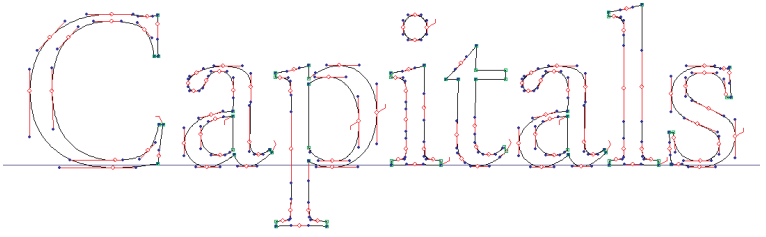
Furthermore there are the options *Y-offset* and *Keep CharacterWidth*.





The Merge Character function can for instance be used to place accents.

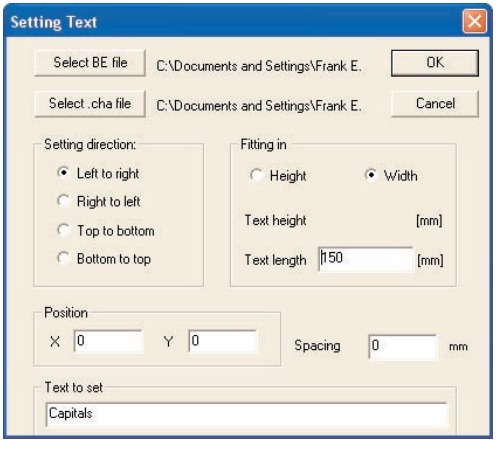
Be aware that to preserve the width of the base character the *Keep CharacterWidth* option must be switched on, otherwise the width of the composite character will differ from the original.



TIP: The merge Character function works only for the active Character Edit Window. To add accents to more than one character the Merge Composites function from Batch menu can be used in combination with a text file that describes which glyphs have to be combined.

Set Text

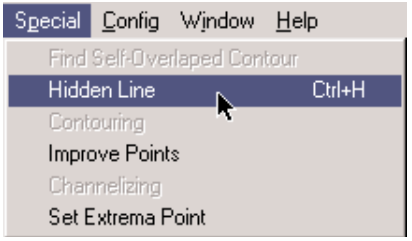
With this function text can be set in the Character Edit Window. Any BE database can be chosen and a range of options is available to control the output. This function is especially useful for the creation of logos.



There are several options in the Setting Text dialog for controlling the output in the Character Edit Window.

SPECIAL MENU

The pulldown menu shows the following options.

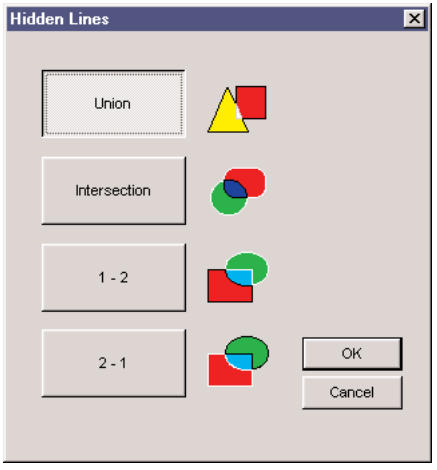


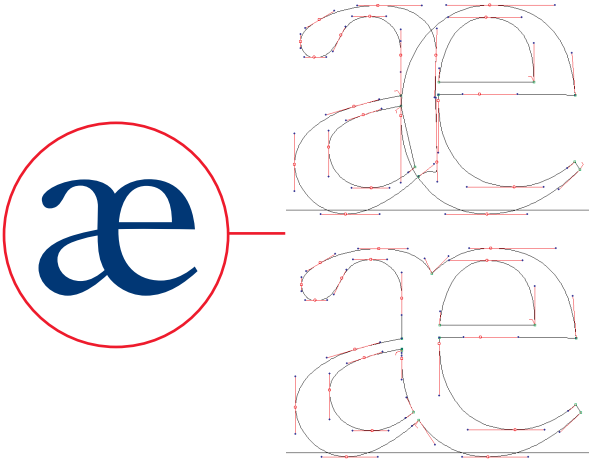
Find Self-Overlapped Contour

The location of this function has been moved into 'Improve points' in this menu.

Hidden Line (⌘ + H) (Ctrl + H)

This function, which is also could have been named *Remove overlap*, merges overlapping contours. It currently works always on the complete character.





The Union option merges the overlapping contours

– *Union*

Merges the contours.

– *Intersection*

Creates the intersection. Only the overlapping parts remain. The rest of the contours are deleted.

– *1-2*

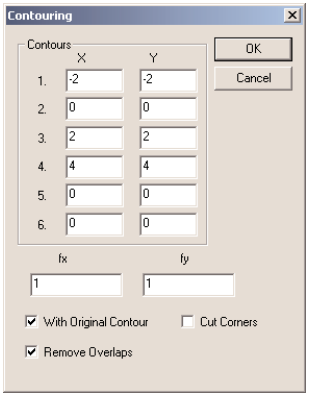
Deletes the second contour and the part of the first contour that was overlapped.

– 2-1

Deletes the first contour and the part of the second contour that was overlapped.

Contouring

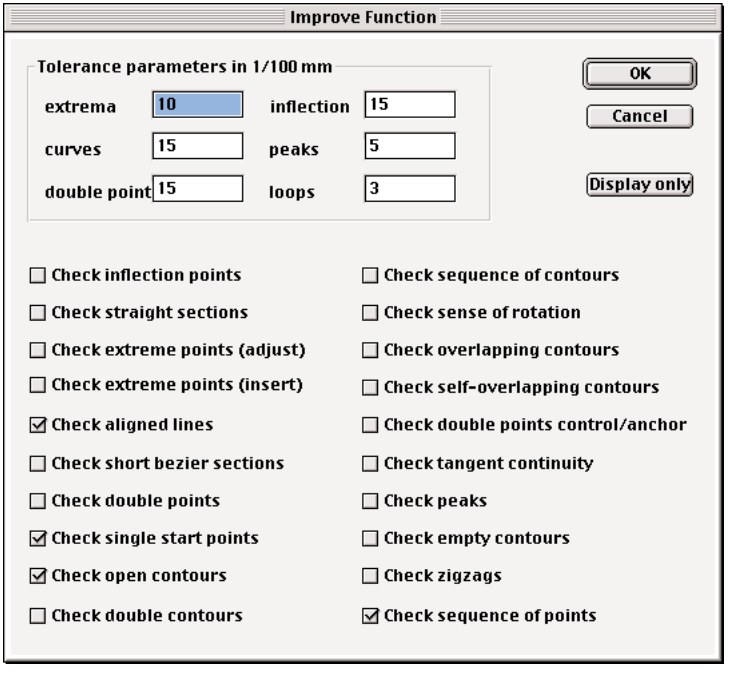
This function is used to create additional contours automatically, so-called outlined characters. It can also be used to bolden a typeface or make it thinner. Input up to 6 values in mm into the menu to create up to 6 additional contours. A positive value of for example 2 adds a contour with 2 mm distance from the original contour to the outside. A negative value works to the inside.

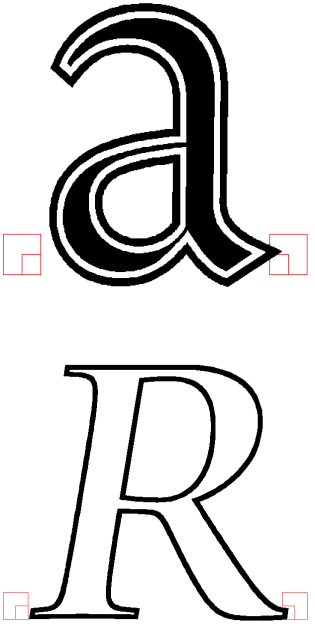


*Up two six contours can be generated using the Contouring function from the **Special** menu.*

Improve Points (⌘ + I) (Ctrl + I)

This function improves the Bezier outline if necessary. Certain problematic digitization features can be removed and corrected, as listed in the popup menu below. You can specify certain parameters which govern the determination of the digitization errors.





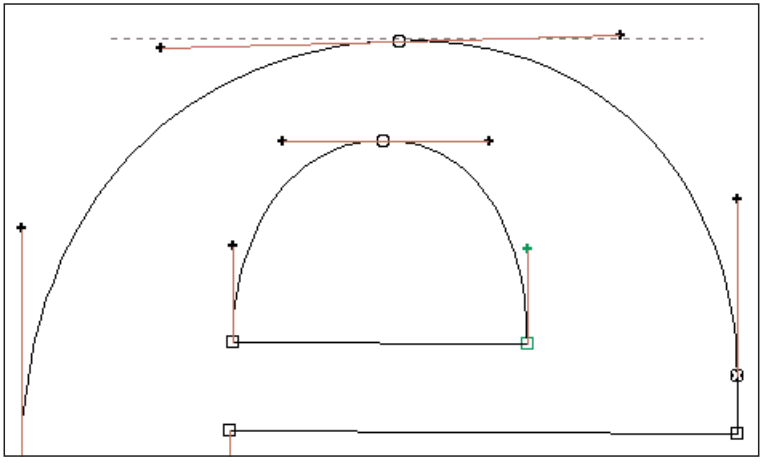
Two examples of contouring. The outline of the capital R was made with different values for the horizontal (slightly thinner) and the vertical lines.

– *Check inflection points*

This function checks whether a Bezier section contains an inflection point or not. It can replace these sections by straight lines if both control points are within the specified tolerance parameters.

– Check straight sections

This function checks whether a Bezier section is nearly a straight line. It can replace these sections by a straight line if both control points are within the specified tolerance parameters.



– Check extreme points (adjust)

This function checks again whether an extreme point on a Bezier curve section is missing. Opposed to the next function it does not add an extreme point but tries to modify the position of the control point in order to make the anchor point an extreme point. If the deviation of the control point position from the horizontal or the vertical is less than the specified tolerance, then the control point will be moved to that position.

– Check extreme points (insert)

This function improves the Bezier outline with respect to missing extreme points. It will insert an extreme point into curve sections which do not have the extreme points as anchor points. The user can specify with the *extrema* parameter the tolerance, which will be the minimal distance for setting the extreme points, i.e. if the new extreme point is closer than the tolerance to one of the anchor points it will not be set.

– Check aligned lines

This function checks whether two consecutive straight sections are nearly parallel and replaces them by one.

– Check short bezier sections

This function works similar to the double point check. It differs in that it also checks curve sections. If the two anchor points of a curve section are closer than the specified tolerance, the complete section will be removed.

TIP: The Improve Points function works only for the active Character Edit Window. To check and improve all characters of a font database DTL ContourMaster can be used. The options in this programme are identical to these in the Improve Points function.

– Check double points

This function checks whether there are control points very close to anchor points. This can be part of the design but it can also indicate a digitization error. Especially if the control point is located on the wrong side of the anchor point the curve can be very problematic for further conversions. If the point is within the specified tolerance, it will be shifted to the position of the anchorpoint.

– Check single start points

This function simply checks whether there are single start points contained in the data, i.e. points which do not belong to a closed contour and which are not connected to any other point. These are typically produced during digitizing and they can lead to conversion errors and raster problems. They will simply be removed by this function. No parameter is needed.

– Check open contours

This function detects open contours. If the correction mode is enabled all contours will be closed. The end point will be shifted to the position of the start point. No parameter is needed.

– Check double contours

This function checks whether there are two identical contours in a character which might have been produced during digitization or editing by accident. The *double point* parameter is used to set the tolerance.

The algorithm works as follows:
– Two contours are considered as identical if they have an identical number of points with an identical nature (anchor/control points) and
– if all pairs of points from the two contours have a distance which is less than the specified tolerance.

– Check sequence of contours

This function reorders the contours according to their hierarchy. Outer contours are ordered before their inner contours. No parameter is needed.

– Change sense of rotation

This function changes the sense of rotation (path direction) of inner and outer contours so that they are opposite and that the black area of the contour is always on the right side. No parameter is needed.

– Check overlapping contours

This function checks whether there are intersections within two contours. No parameter is needed.

– Check self-overlapping contours

This function checks whether there are intersections within one contour. No parameter is needed.

– Check double points Control/Anchor

This function checks whether there are control points very close to anchor points. This can be part of the design but it can also indicate a digitization error. Especially if the control point is located on the wrong side of the anchor point the curve can be very problematic for further conversions. If the point is within the *double point* parameter tolerance it will be shifted to the position of the anchor point.

– Check tangent continuity

This function checks whether two curve sections have tangent continuity at their joining anchorpoint. If the angle is less than 15 degrees the program will make the transition continuous by shifting the control point positions. This function should be used with care and controlled afterwards!

– Check peaks

This function checks whether there are sharp angles within the data. They can be a design feature, but in many cases it indicates a problem. A *peak* parameter is needed.

– Check empty contours

This function checks whether there are contours with area 0 (zero) in the data. They are simply recognized by having less than 4 digitizations. No parameter is needed for this function.

– Check zigzags

This function checks another class of problems which might arise from either digitization errors or from editing mistakes. Zigzags are sequences of anchor points which form acute angles of opposite direction. A *double point* parameter is needed.

– Check sequence of points

This function corrects errors in the sequence of points which prevents correct showing of the characters. This can for instance occur after the interpolation of BE databases.

CONFIG MENU

Function Settings: Parameters

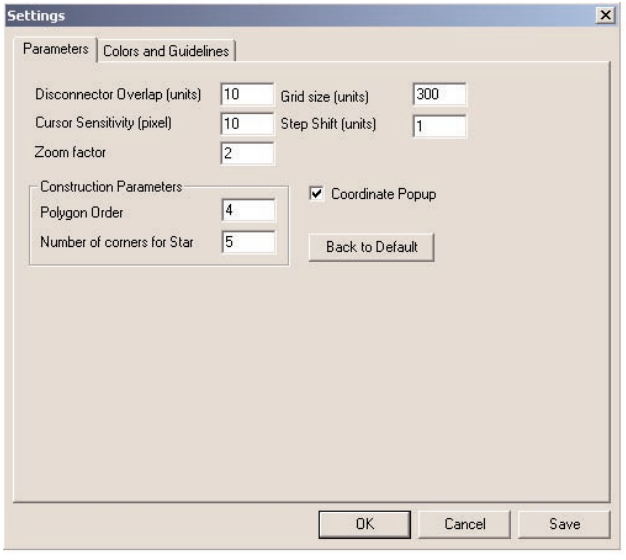
In the **Config** menu *Function Settings: Parameters* you can set different parameters:

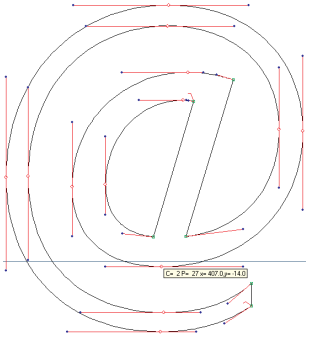
Function	*Default value*
Disconnector Overlap	100 (units, 1/100 mm)
Cursor Sensitivity	10 (units, 1/100 mm)
Zoom	2 (factor)
Grid size	300 (units, 1/100 mm)
Step Shift	15 (units, 1/100 mm)

Please adjust these parameters if the bodysize of your font is not 15000 (units), which is the default. The 15000 units system is used here because of the fact that DTL BezierMaster is originally based on the Ikarus system. The basic idea behind this system is, that a high resolution of the database means a greater control over the details in the design and also a relatively small loss of quality when the resolution is scalled down for the generation of PostScript Type 1 and OpenType (CFF) fonts (1000 units) or TrueType fonts and OpenType (TTF) fonts (2048 units). However, an EM-square of 1000 units is standard for the Bezier format. For instance the *Step Shift* should be one then.

The construction parameters determine the number of corners of a polygon and a star.

If the *Coordinate popup* option is enabled, then the coordinates of the point nearest to the cursor will be shown in the Character Edit Window.





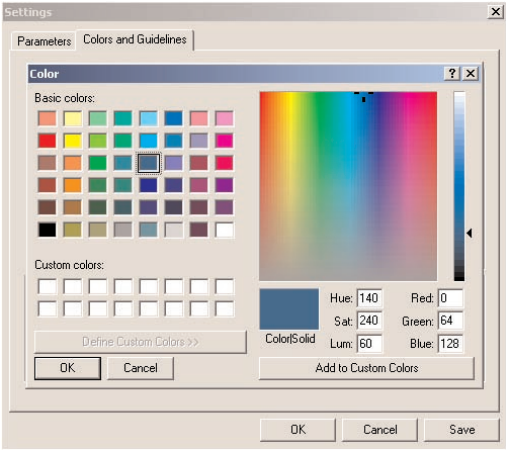
If the Coordinate popup option is enabled, the coordinates of the point nearest to the cursor will be shown.

Function Settings: Color

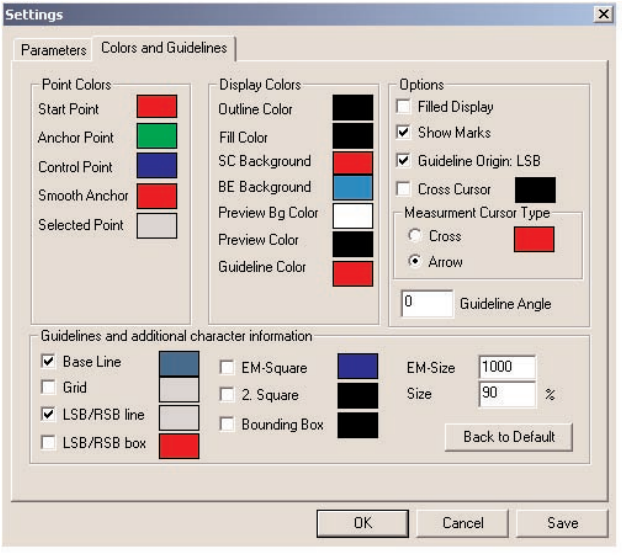
In the **Config** menu *Settings: Color* you can set the colors and the guidelines which shall be shown or not.
The meaning is mostly self-explaining:
X minimum / X maximum
Y minimum / Y maximum
Base Line (at Y position: 0)
Grids (Grid distance can be set in the Parameters menu above)
LSB / RSB line = Left side bearing / Right side bearing
EM-size (standard BE-bodysize = 15000 (1/100 mm))
Cross Cursor (shows a crosshair cursor instead of an arrow)

Marks: Colors (default)
Start point *red*
Corner point = Anchor point *green*
Curve point = Control point *blue*
Tangent point = Smooth Anchor point *red*
Outline color *black*
SC background *red*
Selected point *grey*
Fit to guide *grey*
Self-overlap *grey*
BE background *yellow*
Preview Text Color *black*
Preview Background Color *white*

Just double click on the colors to change them.



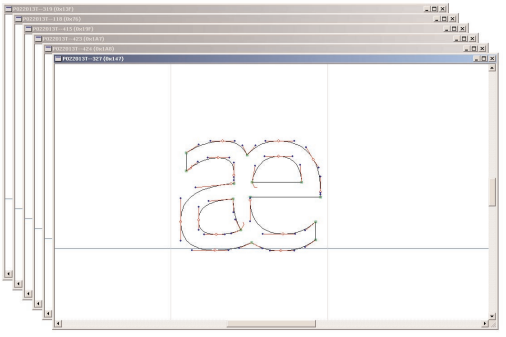
Arbritary colors can be selected.



WINDOW MENU

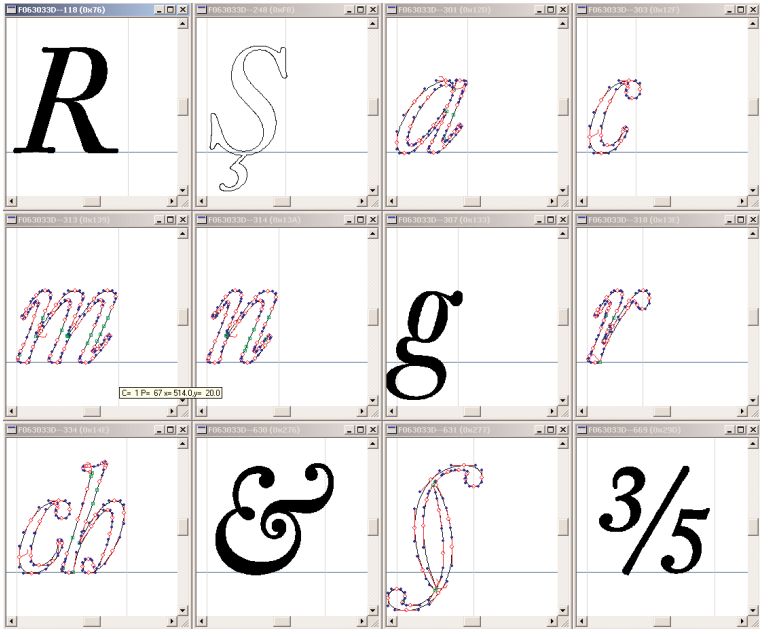
Cascade

When more than one Character Edit Window is opened, they overlap each other. By choosing *Cascade* they are ordered in cascading windows that can be moved to the foreground by clicking them.



Tile (⌘ + J) (Ctrl + J)

By choosing *Tile* all opened Character Edit Windows are placed in a tile pattern.



All open characters are placed in a tile pattern.

Close all windows

When this function is chosen, all Character Edit Windows are closed. The Character List Window remains opened.

Function	Mac OS	Windows
Arrow	(Space) (⌥Alt + A)	(Space) (Alt + A)
Background on/off	(Ctrl + B)	(Alt + B)
Background chars on/off	(⌘ + B)	(Ctrl + B)
Change Character Header	(⌘ + I)	(Ctrl + I)
Change Font Header	(⌘ + ⇧Shift + F)	(Ctrl + ⇧Shift + F)
Change Label	(⇧Shift + 3)	(⇧Shift + 3)
Close	(⌘ + W)	(Ctrl + W)
Copy	(⌘ + C)	(Ctrl + C)
Copy into Background	(⌘ + Ctrl + C)	(Ctrl + Alt + C)
Cut	(⌘ + X)	(Ctrl + X)
Delete character	(← Backspace)	(Delete)
Delete Point	(⇧Shift + I)	(⇧Shift + I)
Digitizing Number	(⌥Alt + D)	(Alt + D)
Display Marks	(⌘ + M)	(Ctrl + M)
Edit Coordinates	(⌘ + E)	(Ctrl + E)
EM Square on/off	(⌘ + D)	(Ctrl + D)
Exit	(⌘ + Q)	(Ctrl + Q)
Fill Color	(⌘ + F)	(Ctrl + F)
Font Administration	(⌘ + U)	(Ctrl + U)
Grids	(⌘ + G)	(Ctrl + G)
Hidden Line	(⌘ + H)	(Ctrl + H)
Horizontal Guidelines	(⇧Shift + H)	(⇧Shift + H)
Improve points	(⌘ + I)	(Ctrl + I)
Insert Point	(⇧Shift + 2)	(⇧Shift + 2)
Measurement	(Ctrl + S)	(Alt + S)
Metrics Editor	(⌘ + K)	(Ctrl + K)
Move Screen	(W)	(W)
New	(⌘ + N)	(Ctrl + N)
Next Character	(⌘ + →KeyRight)	(Ctrl + →KeyRight)
Open	(⌘ + O)	(Ctrl + O)
Paste (with offset)	(⌘ + V)	(Ctrl + V)
Paste (without offset)	(⌘ + ⇧Shift + V)	(Ctrl + ⇧Shift + V)
Paste from Background	(⌘ + Ctrl + V)	(Ctrl + Alt + V)
Pen Tool	(Ctrl + Q)	(Alt + Q)
Previous Character	(⌘ + ←KeyLeft)	(Ctrl + ←KeyLeft)
Print	(⌘ + P)	(Ctrl + P)
Print Options	(⌘ + ⌥Alt + P)	(Ctrl + Alt + P)
Print Setup …	(⌘ + ⇧Shift + P)	(Ctrl + ⇧Shift + P)
Redo	(⌘ + Y)	(Ctrl + Y)

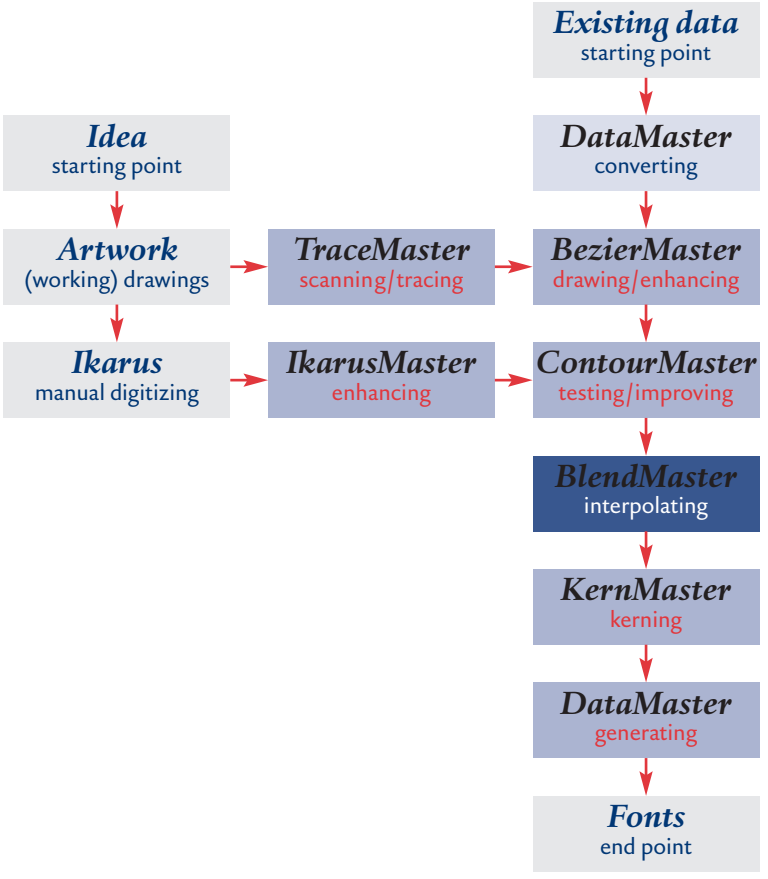
Function	Mac OS	Windows
Replace by Background	(⌘ + Ctrl + R)	(Ctrl + Alt + R)
Reset	(⌘ + R)	(Ctrl + R)
Rotate	(Ctrl + Z)	(Alt + Z)
Save	(⌘ + S)	(Ctrl + S)
Save as	(⌘ + ⇧Shift + S)	(Ctrl + ⇧Shift + S)
Scaling	(Ctrl + W)	(Alt + W)
Second EM Square on/off	(⌘ + ⇧Shift + D)	(Ctrl + ⇧Shift + D)
Select all points	(⌘ + A)	(Ctrl + A)
Select Background	(⌘ + ⇧Shift + B)	(Ctrl + ⇧Shift + B)
Shift smooth	(Ctrl + X)	(Alt + X)
Tile	(⌘ + J)	(Ctrl + J)
Undo	(⌘ + Z)	(Ctrl + Z)
Vertical Guidelines	(⇧Shift + V)	(⇧Shift + V)
V/H Guide Lines	(⌘ + ⇧Shift + L)	(Ctrl + ⇧Shift + L)
Zoom +,–	(E)	(E)

The Digital Font Collection of the Dutch Type Library, containing more than 300 fonts in PostScript Type 1 and TrueType format for Mac OS and Windows and, of course, in OpenType format.

Dutch Type Library

DTL BlendMaster

's-Hertogenbosch / Hamburg
Summer 2004



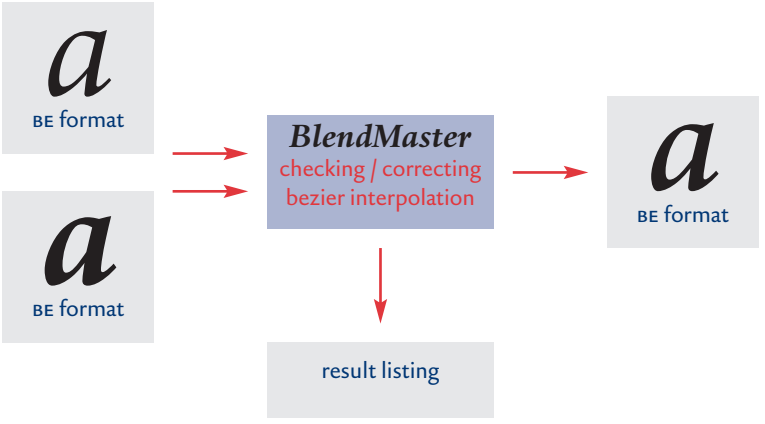
The diagram shows a typical workflow based on the modules of DTL *FontMaster.*

DTL BlendMaster is a module to interpolate (blend) fonts. Normally, a medium (also called semibold) is calculated from a regular and a bold variant of a typeface, which together form the poles. Seen purely mathematically, interpolating is not an especially complicated technique, but the blending of fonts is not easy. DTL BlendMaster checks and corrects the contours of the poles in such a way that interpolation ultimately proceeds in a checked and faultless manner.

DTL BlendMaster uses intelligent routines, which make the number of points, extreme points, start points and the direction of the contours of both poles equal without changing the form of the glyphs.

Function description: Bezier interpolation

The interpolation of two BE fonts will require two isomorphic fonts. The data flow is quite simple and looks like this:



The interpolation requires only a few parameters. The interpolation factor can be independently defined for X-factor an Y-factor. Normally the range for blending is between 0 and 100 percent.

Multiple file acces (batch)

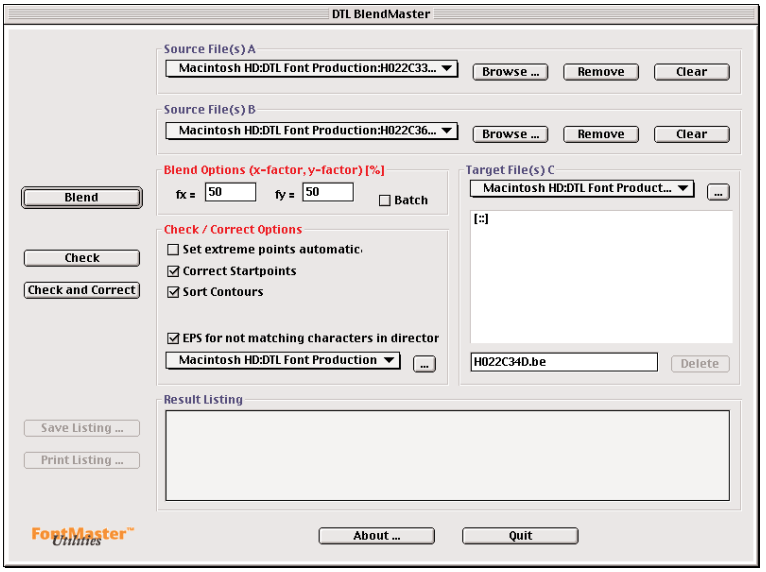
It is possible to select multiple files from two different folders and to process them with one command. The two folders should contain the same number of files. The output file names will be generated automatically by appending the factor to the font names. If the factor is 50 for example then '_IN5000' will be added. When the interpolation factor is 75 percent then the font name will look like this: A057C13T_IN7500.be

Starting DTL BlendMaster

The program dialog has four major functions:
–To select one or more source files.
–To select one or more options for checking and correcting the source fonts.
–To define the parameters for the blending of the source fonts.
–To trigger the checking, correcting or blending.

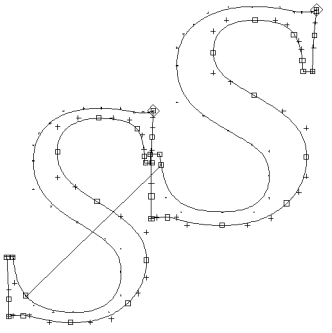
1. Blend

Before fonts can be interpolated the *Source Files* have to be selected. The font entries here can be changed by removing fonts separately or clearing the font list completely. If two fonts are selected for blending, the font



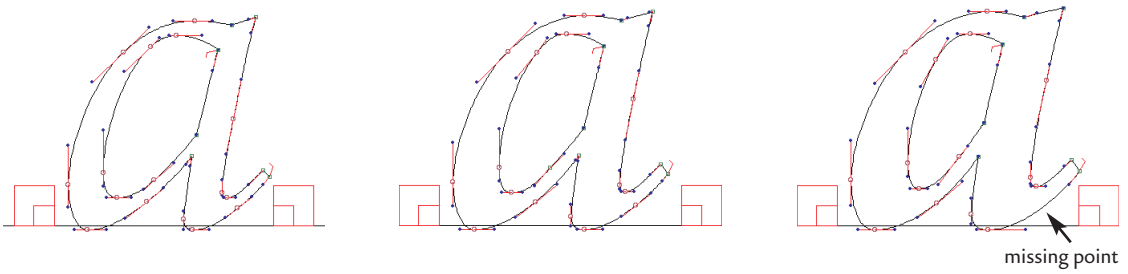
If characters fail to blend, an EPS showing the problem(s) can be generated. It is possible to import this EPS in DTL BezierMaster.

name from *Source File(s) A* will be the basis for the name of the resulting font in the *Target File(s) C* folder. Before blending can take place, the X-factor and Y-factor should be defined. Characters that fail to blend can be saved as Encapsulated PostScript files that will show where the problem(s) occurred. This way it is possible to make the necessary changes in the contours by hand in, for instance, DTL BezierMaster. For this the option *EPS for non matching characters in directories* must be selected and a target directory for the EPS files must be chosen.



Interpolation
Interpolation
Interpolation

The regular version (top) and bold version (bottom) of DTL Unico were blended with X- and Y-factor of 50% to create the medium weight (center).



Differences on point level between the two versions of the letter are automatically corrected to make blending, resulting in the a in the center, possible.

2. Check

Before blending the fonts, they can be checked. Differences on point level are detected by default. Furthermore there are three options:

–*Set extreme points automatically*

–*Correct Startpoints*

–*Sort Contours*

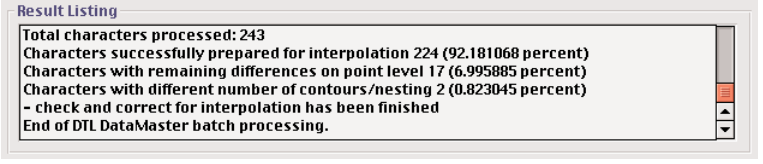
Normally these three options should be activated. If the Batch function is selected, the *Check* function will be disabled.

The result of the check is displayed in the *Result Listing* window and will look like this:

```
DTL DataMaster batch processing is being started.
* check options:
- Correct startpoints
- Sort contours
- EPS for not matching characters in directory "Macintosh HD:DTL Font Production"
+ check and correct for interpolation is being started
- source and target file A is "Macintosh HD:DTL Font Production:Batch_1:A057C33T.be"
- source and target file B is "Macintosh HD:DTL Font Production:Batch_2:A057C36T.be"
Processing character 101 (A)
correction successfull
Processing character 102 (B)
correction successfull
Processing character 103 (C)
correction successfull
Processing character 104 (D)
differences on point level detected (diff num of digs)
correction successfull
Processing character 105 (E)
correction successfull
Processing character 106 (F)
correction successfull
Processing character 107 (G)
differences on point level detected (diff num of digs)
correction successfull
Processing character 108 (H)
correction successfull
Processing character 109 (I)
correction successfull
Processing character 110 (J)
```

3. Check and Correct

It is standard procedure to choose this function before blending. The contours are prepared this way for optimal interpolation. A listing will show information about the percentage of succesfully prepared characters, the remaining differences on point level and the number of characters with a different number of contours.



Beware of the fact that changes to the contours are automatically saved in the files that serve as basis for the blending. The changes are irreversible; it is recommended to use copies of the original files for blending.

4. Save Listing ...

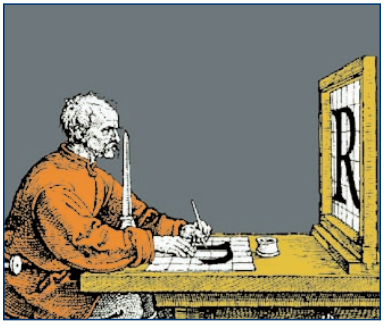
Listings can be saved as text files.

5. Print Listing ...

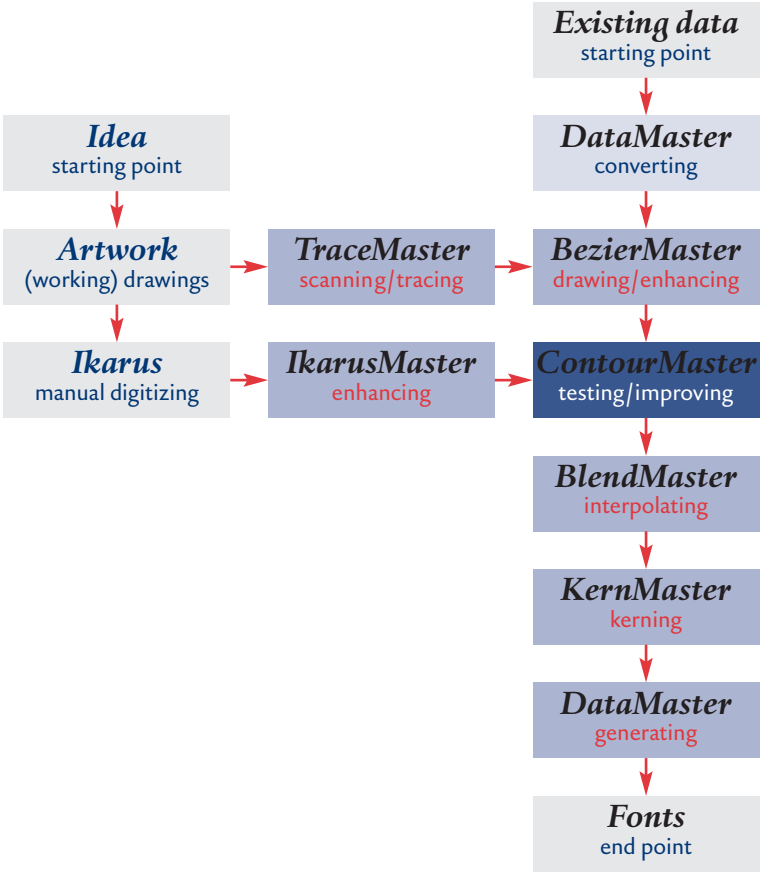
Listings can also be printed directly.

Dutch Type Library

DTL ContourMaster



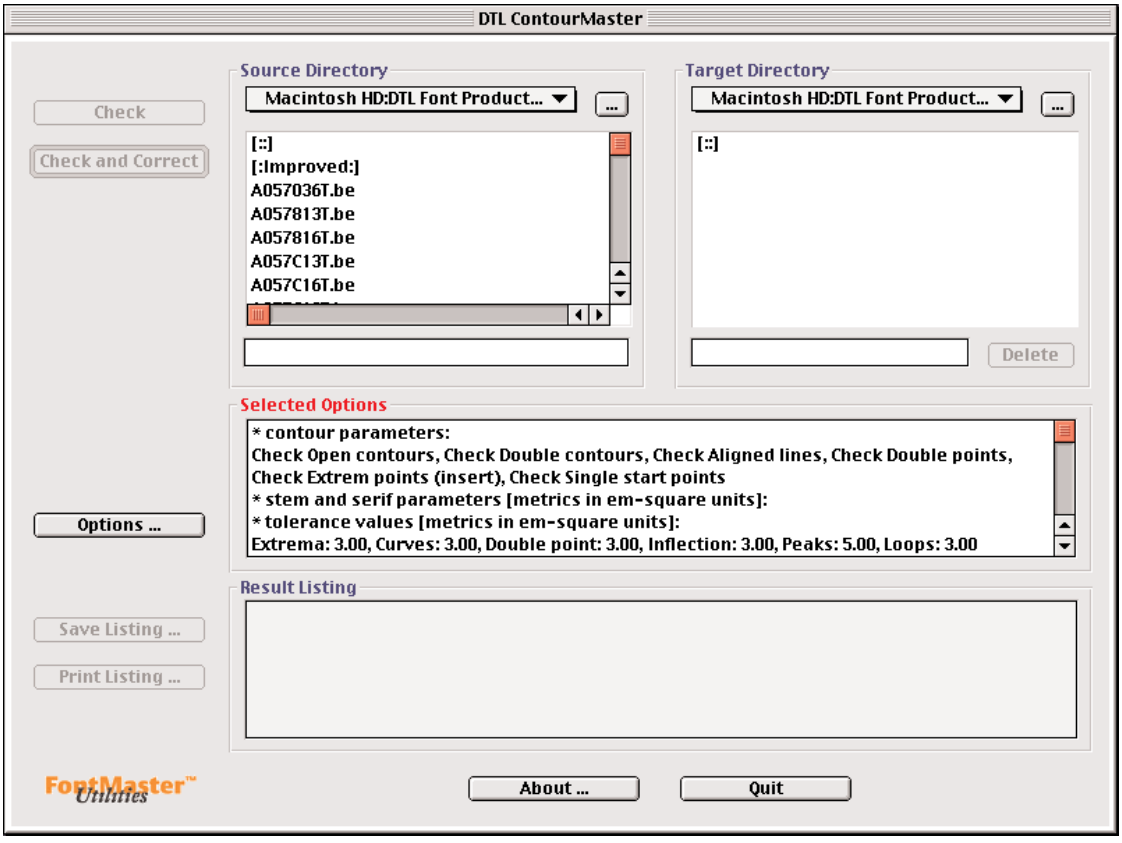
's-Hertogenbosch / Hamburg
Summer 2004



The diagram shows a typical workflow based on the modules of DTL *FontMaster.*

DTL ContourMaster was the first module that was used by the Dutch Type Library. The program can be used to test and correct all kinds of errors in contours that are made in sizable productions in which several people participate. Typical errors are open contours, double points, overlapping contours, double contours and missing extreme points. Things like bar thickness and line thickness can also be tested.

A program like DTL ContourMaster is indispensable, especially for large quantities of characters that together form several codepages in a TrueType or OpenType font. DTL ContourMaster tests the BE format, which was developed by URW++ and derived from the Ikarus format. This enables a description of contours at extremely high resolution.



Starting DTL ContourMaster

The first dialog has two major functions:

– To select one or more source font(s).

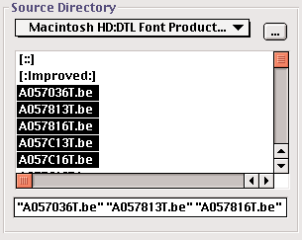
– To select one or more options for checking and correcting the source font(s).

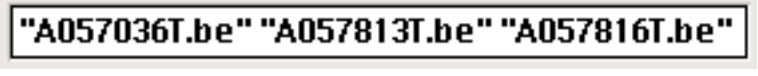
1. Check

Before fonts can be checked a series of selections have to be made, such as the selection of the Source Directory, the Target Directory and the Basic Font Options. The selected options determine what will be checked.

1.1 *Source Directory*

This directory contains the font data to be checked. The button marked '...' shows the standard system dialog for browsing. All the fonts in the BE format will be listed and can be selected with the mouse. The selected fonts are shown below.



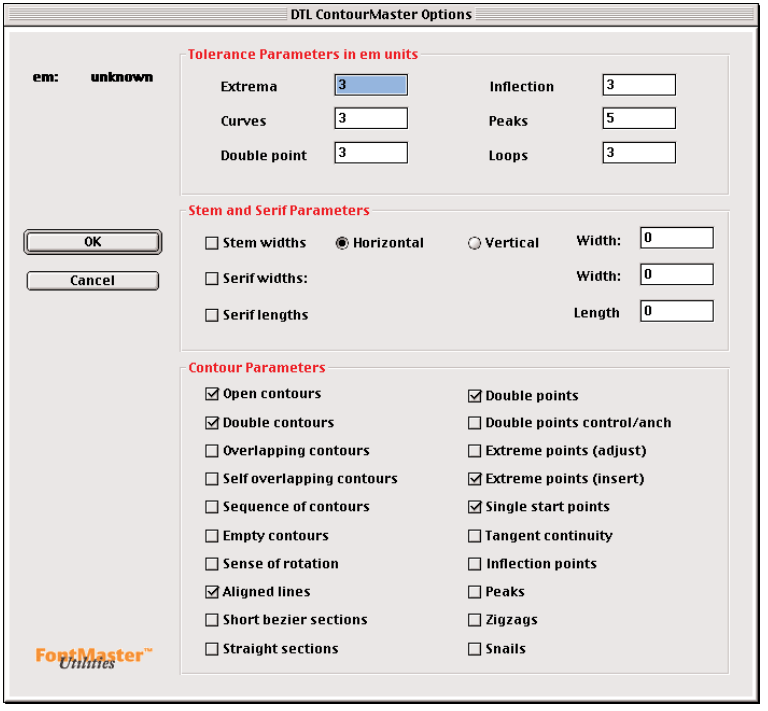


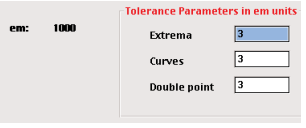
1.2 *Target Directory*

If the fonts are only checked, the Target Directory is not used because the font data will not be altered.

2. Check and Correct

This function automatically corrects the contours, based on the chosen options. Changes to the contours are made automatically and the changes are irreversible. It is advisable to make a copy first, so that the original and the improved version can be compared in DTL BezierMaster.





If no font is selected when the Options dialog is opened, the value of the EM will be unknown. Otherwise the EM will be shown.

3. Options

A series of options is available and pressing the '**Options …**' button opens the **DTL ContourMaster Options** dialog.

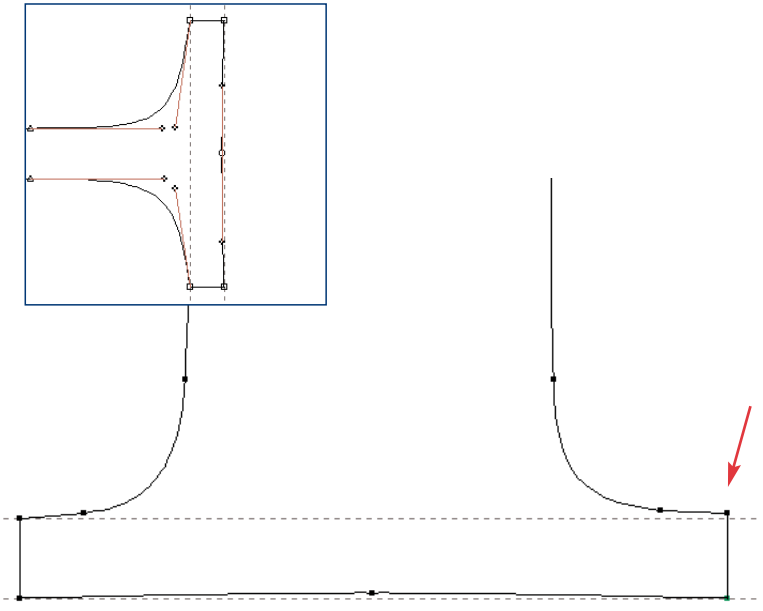
A number of parameters can be entered here:
– *Tolerance Parameters in EM units*
These settings influence the Contour Parameters. Deviations smaller than the given tolerances will be neglected.
– *Stem and Serif Parameters*
Stem widths, serif widths and lengths can be checked only and are compared with the defined values.

3.1 *Stem widths*

This function checks and corrects the width of the horizontal and vertical stems according to the entered value.

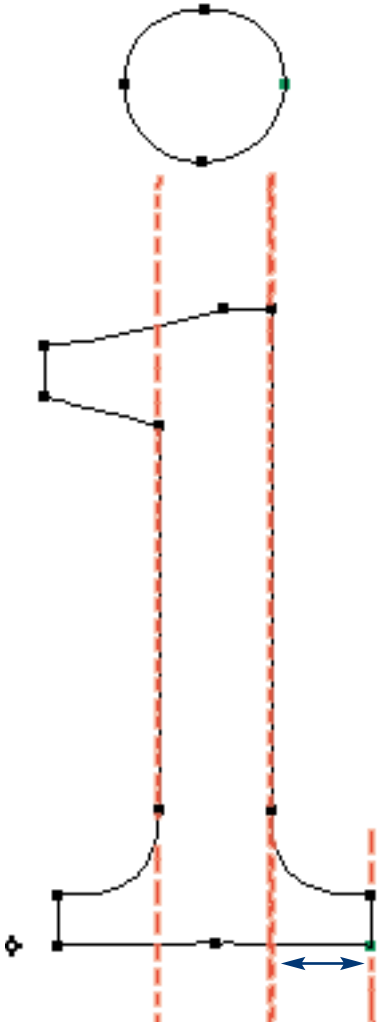
3.2 *Serif widths*

Checks and corrects the width of the horizontal and vertical serifs according to the entered value.



3.3 *Serif lengths*

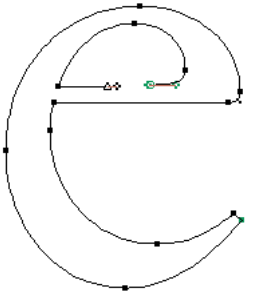
This function checks and corrects the length of the horizontal serifs according to the entered value.



Beside stem widths as well serif lengths can be corrected

3.4 *Open contours*

This function detects open contours. If the correction mode is enabled all contours will be closed. The endpoint will be connected with the startpoint.

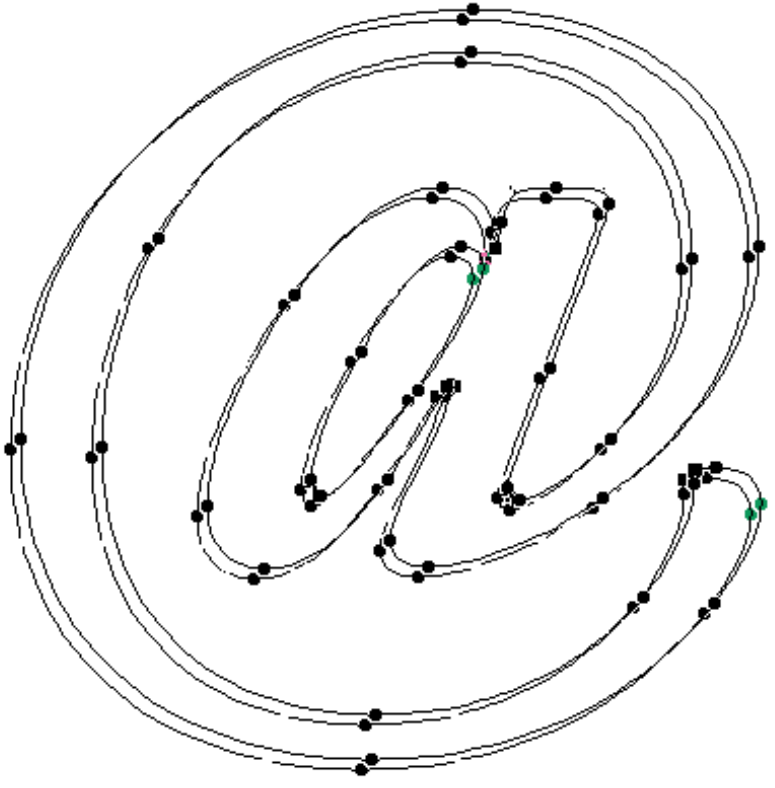


In case of an open contour the start- and endpoint are connected.

3.5 *Double contours*

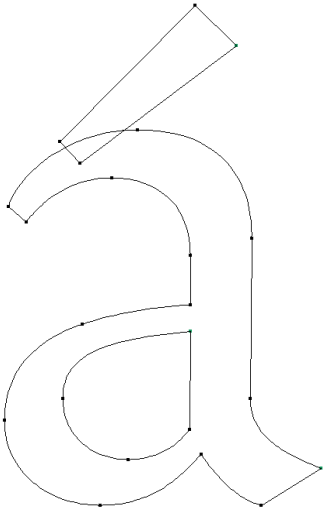
Checks whether there are two identical contours in a character which might have been produced during digitization or editing by accident.

The algorithm works as follows:
– Two contours are considered identical if they have an identical number of points with identical nature (anchor/control points) and
– If all pairs of points on the two contours have a distance which is less than the tolerance which is specified.

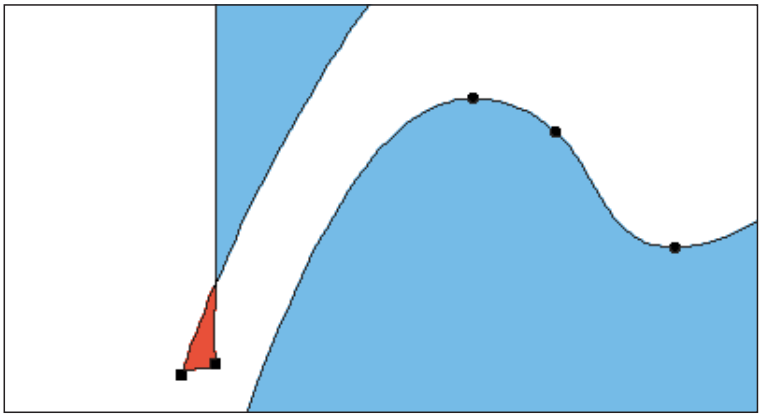


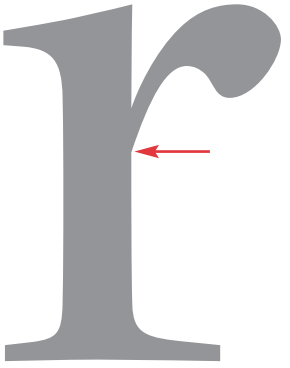
3.6 *Overlapping contours*

This function checks whether there are intersections within two contours.



Overlapping contours.





3.7 *Self overlapping contours*

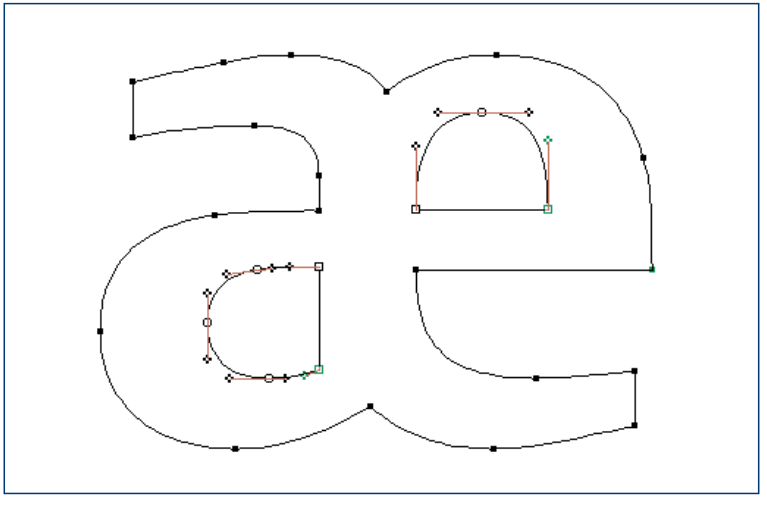
Checks whether there are intersections within one contour.

3.8 *Sequence of contours*

Reorders the contours according to their hierarchy. Outer contours are ordered before their inner contours.

3.9 *Empty contours*

This function checks whether there are contours with area 0 (zero) in the data. They are simply recognized by having less than 4 digitizations.



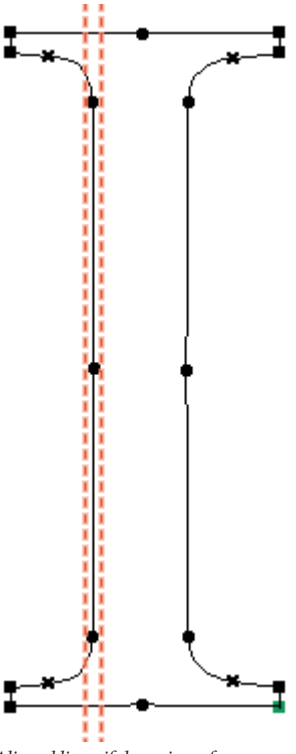
3.10 *Sense of rotation*

This function changes the sense of rotation (path direction) of inner and outer contours so that they are opposite and that the black area of the contour is always on the right side.

3.11 *Aligned lines*

This function checks whether two consecutive straight sections are nearly parallel and replaces them by one.

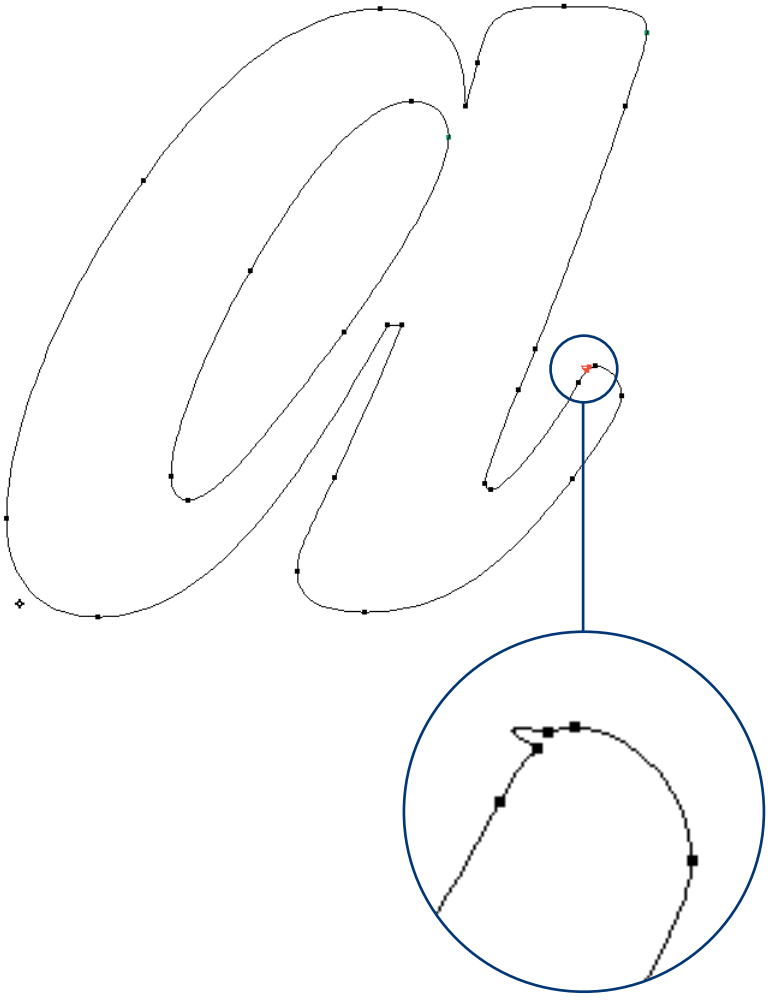
If the two straight sections are very close within the tolerance to the connecting line as indicated by the dashed lines the middle point will be removed.



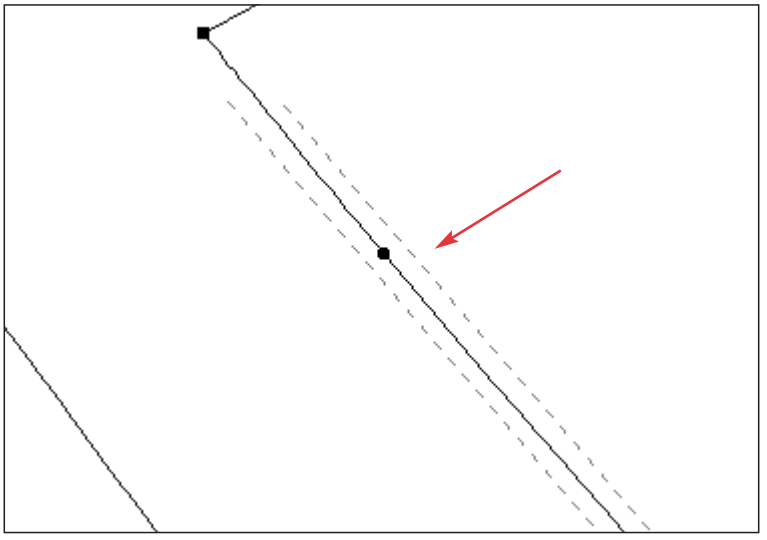
Aligned lines: if the points of two straight sections are within the specified tolerance (indicated by the dashed lines) the middle point will be removed.

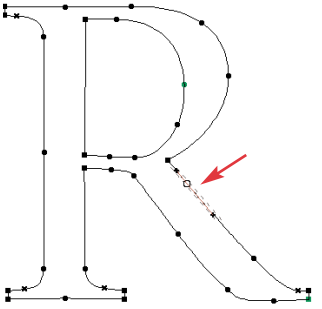
3.12 *Short bezier sections*

This function works similar to the double point check. Different from that function it checks also curve sections. If the two anchor points of a curve section are closer than the tolerance then the complete section will be removed.



Short bezier sections: very narrow anchor points. The complete section will be removed if its points are within the tolerance.





This bezier section is nearly straight. The channel defined by the tolerance is indicated with dashed lines. If both control points are within this channel, the Bezier curve will be changed to a straight line.

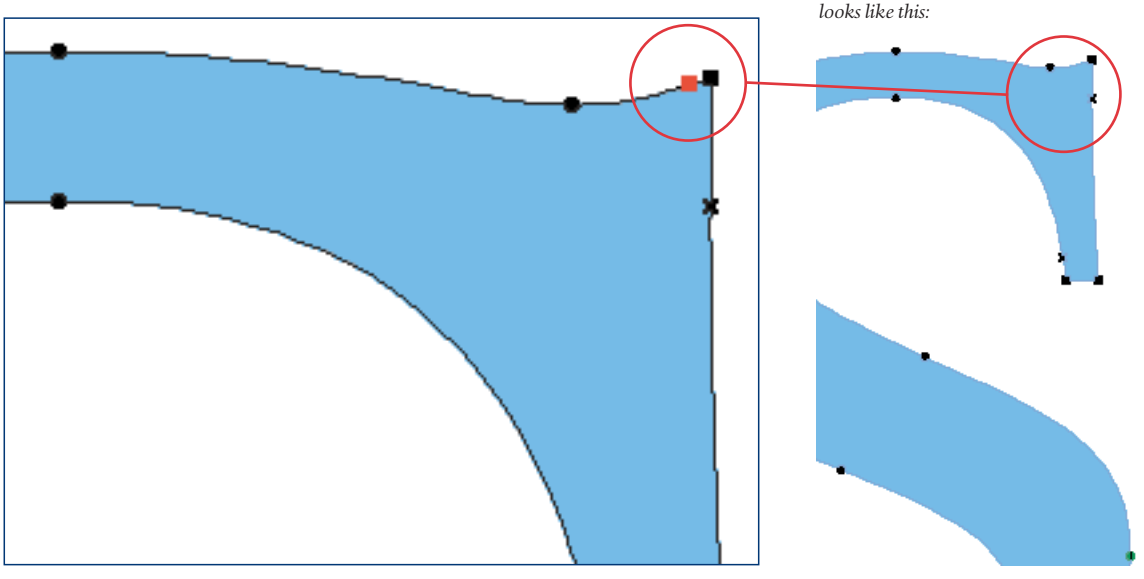
3.13 *Straight sections*

This function checks whether a Bezier section is nearly a straight line. It can replace these section by a straight line if both control points are within the specified channel tolerance.

3.14 *Double points*

This function is the classical check for double points. If the distance of two neighboured anchor points is less than the tolerance specified, the second point will be deleted.

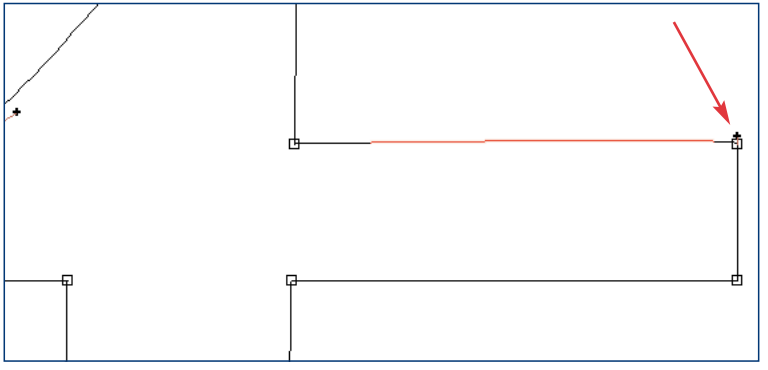
After the point is deleted, the contour looks like this:



3.15 *Double points control/anchor*

This function checks whether there are control points very close to anchor points. This can be a design feature, but it can also indicate a digitization error. Especially if the control point is located on the wrong side of the anchor point the curve can be very problematic for further conversions.

If the point is within the tolerance, it will be shifted to the position of the anchor point.

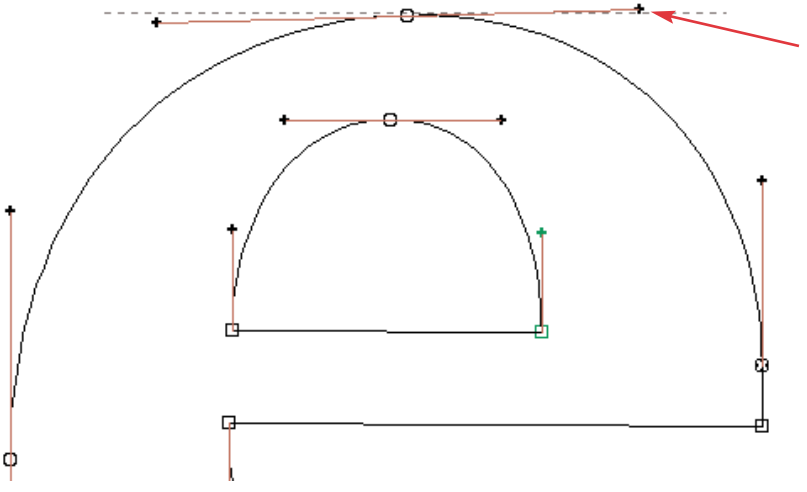


The arrow indicates a problematic control point position.

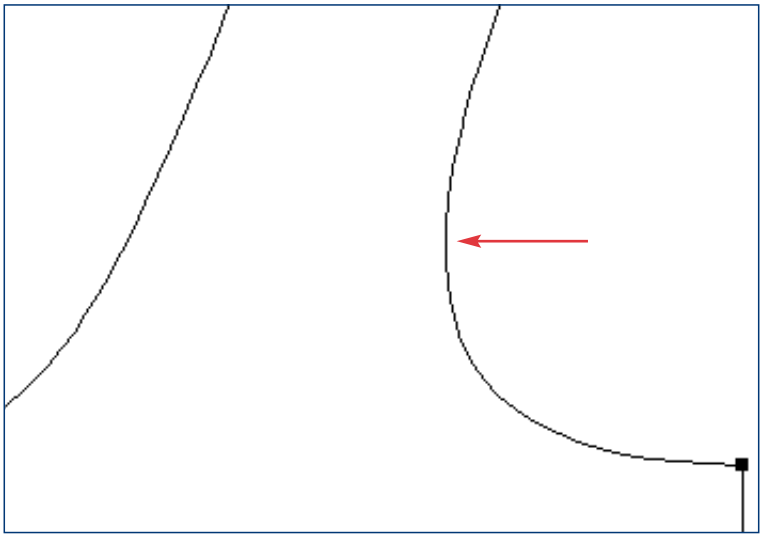
3.16 *Extreme points (adjust)*

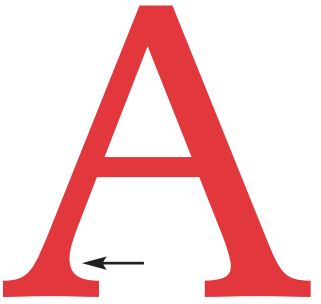
This function checks again whether an extreme point on a Bezier curve section is missing. In contrast to the other function, it does not add an extreme point but tries to modify the position of the control point in order to make the anchor point an extreme point.

If the deviation of the control point position from the horizontal or vertical is less than the specified tolerance, then the control point will be moved to that position.



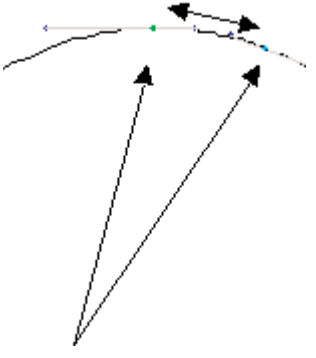
The control point is not horizontal. If it deviates less than the tolerance, then it will be aligned horizontally.



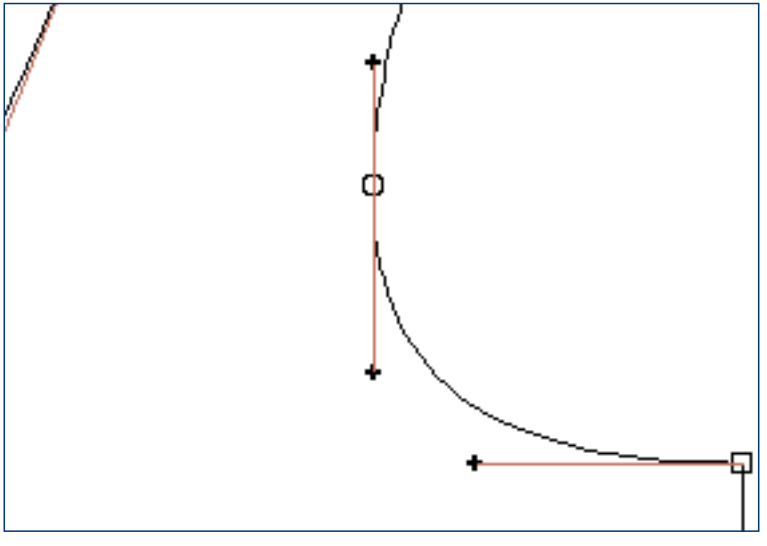


3.17 *Extreme points (insert)*

This function just sets the extreme points. It will set extreme points into curve sections which do not have the extreme points as anchor points. The user can specify a distance which will be the minimum distance for setting the extreme points, i.e., if the new extreme point is closer than the tolerance to one of the anchor points, it will not be set.



Extreme point and next anchor point.

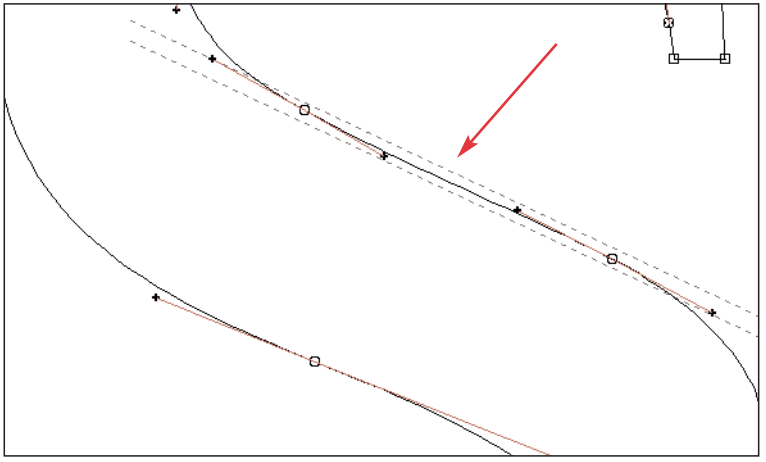


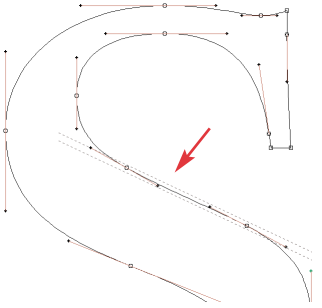
3.18 *Single start points*

This function simply checks whether there are single starting points in the data, i.e. points which do not belong to a closed contour and which are not connected to any other point. These are typically produced during digitizing and they can lead to conversion errors and rasterizing problems. They will simply be removed by this function.

3.19 *Tangent continuity*

Checks whether two curve sections have tangent continuity at their joining anchor point. If the angle is less than 15 degrees the program will make the transition continuous by shifting the control point positions. This function should be used with care and should be checked afterwards.





This bezier section contains an inflection point. The channel defined by the tolerance is indicated with dashed lines. If both control points are within this channel the Bezier curve will be changed to a straight line.

3.20 *Inflection points*

This function checks whether a Bezier section contains an inflection point or not. It can replace these sections by straight lines if both control points are within a channel tolerance specified.

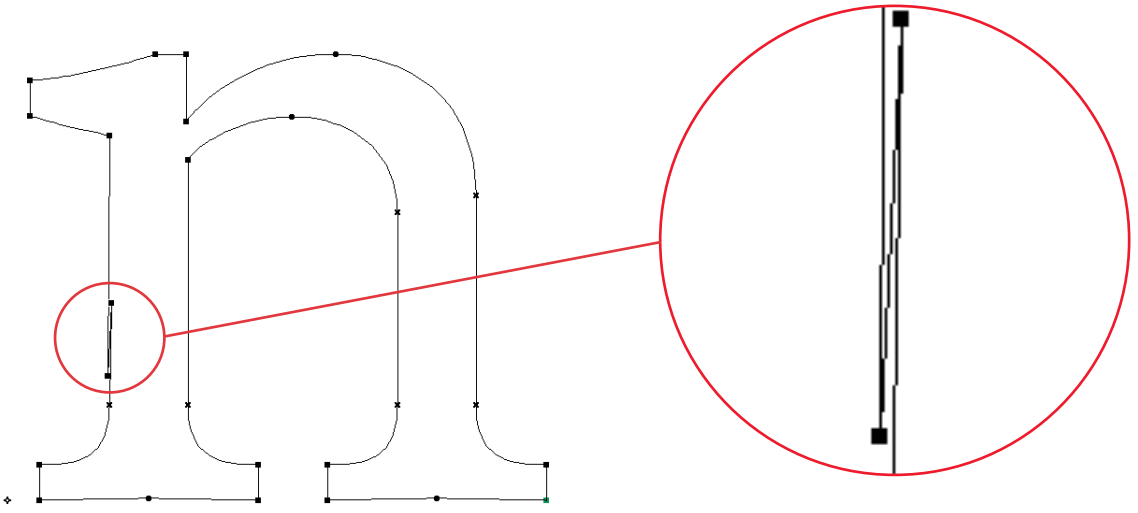
3.21 *Peaks*

Checks whether there are sharp angles within the data. They can be due to design but in many cases it indicates a problem. These peaks are simply reported and not corrected.

3.22 *Zigzags*

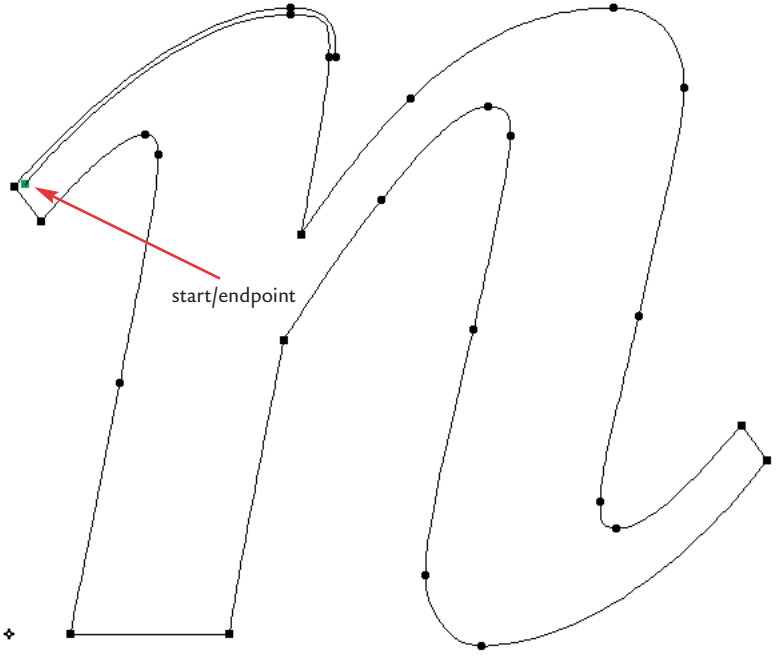
Checks another class of problems which might arise from either digitization errors or from editing mistakes. Zigzags are sequences of anchor points which form sharp angles of opposite direction.

If the two points are corners with a sharp angle (larger 90 degree) and are within the tolerance to the connecting line, these points will be removed.

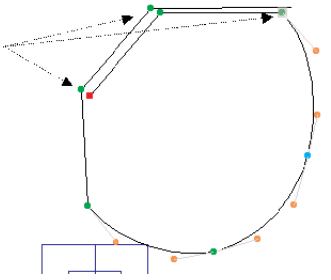


3.23 *Snails*

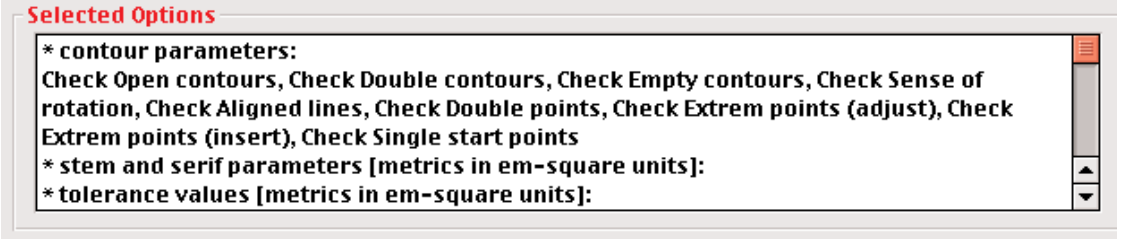
This function again is used to check digitization errors. A snail is a sequence of points on top of each other which exceeds the endpoint. It is an open contour by definition.



This error can be detected and removed but a tolerance value is necessary. The algorithm works as follows:
– The program checks whether the contour is open.
– The program checks whether there is a point close to the end point.
– The programs checks whether all points after the end point close to the end point are close to another digitization.



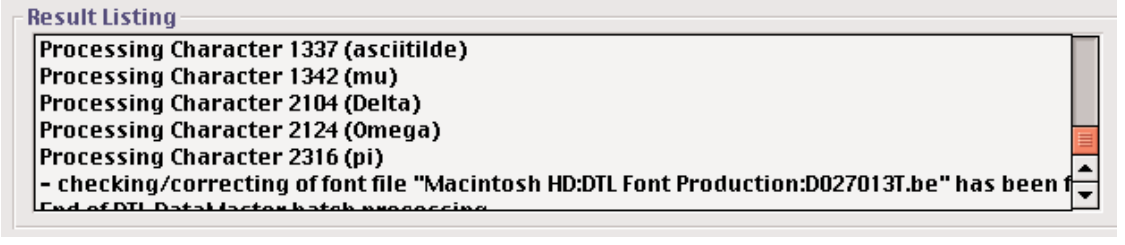
Close to another point means within a radius less than the specified tolerance.



The selected options are listed in the DTL ContourMaster dialog.

4. Save Listing …

After checking and/or correcting the results of the actions will be listed.



This listing can also be saved as a text file.

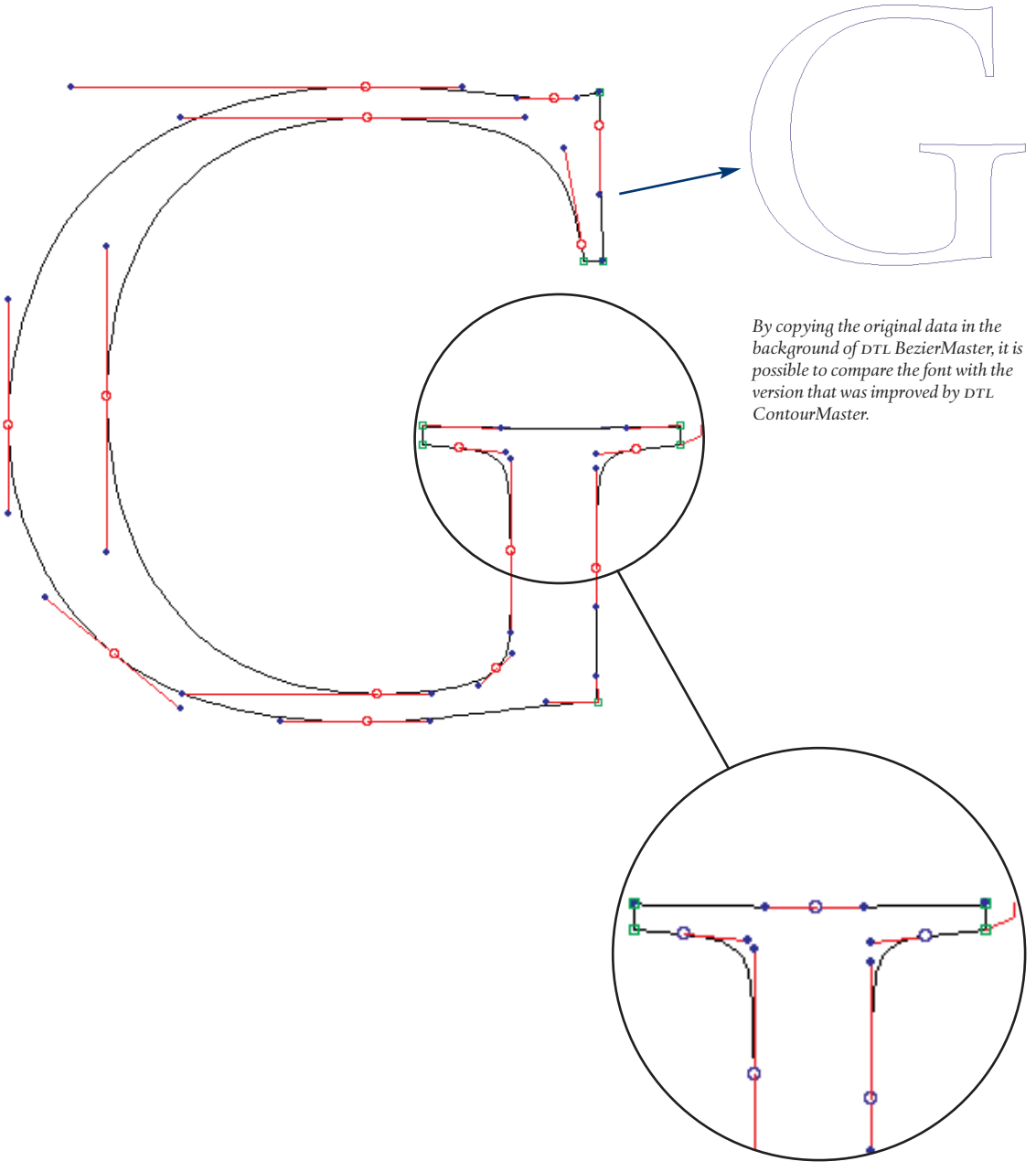
```
+ checking font file “Macintosh HD:DTL Font Production:D027013T.be” and
writing corrections to “Macintosh HD:DTL Font Production:Improved:D027013T.be”
* em-square is 1000, metrics data refer to this value
Processing Character 101 (A)
Processing Character 102 (B)
Processing Character 103 (C)
extrem point violation at dig 11, dphi -94.811708 [deg]
    anchor point adjusted, dig 13
Processing Character 104 (D)
Processing Character 105 (E)
```

Part of a saved listing.

5. Print Listing …

This function sends the listing directly to the printer.

Corrections made by DTL ContourMaster are irreversible. To protect your original data it is not possible to overwrite the 'master' file. After your font data is optimized, it is of course possible to use DTL BezierMaster to compare the original with the new version by opening one font in the foreground and selecting the other one in the background.



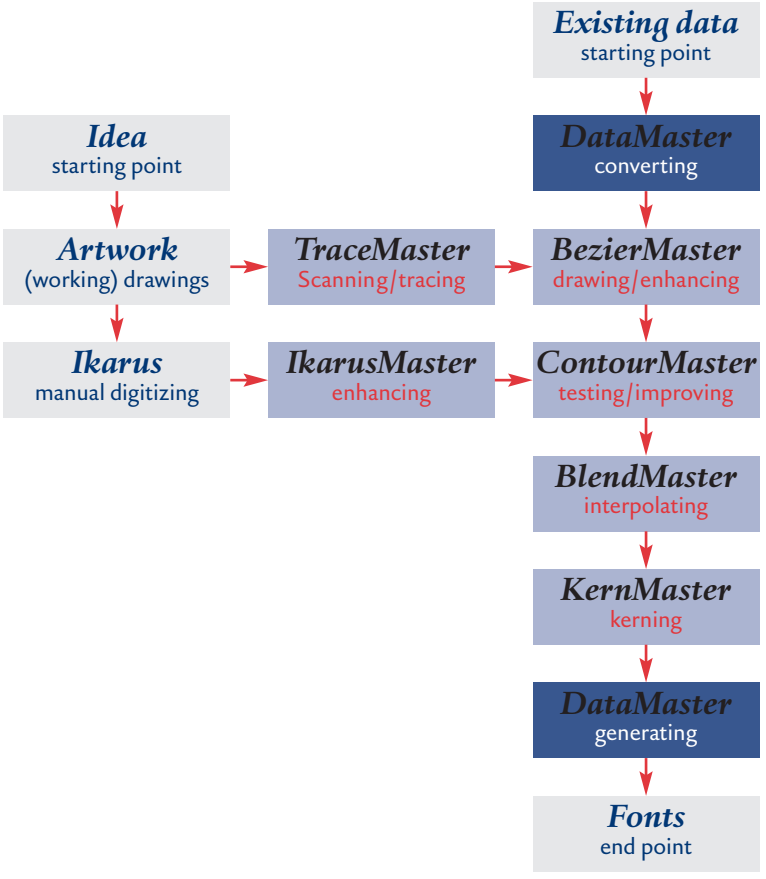
By copying the original data in the background of DTL BezierMaster, it is possible to compare the font with the version that was improved by DTL ContourMaster.

Matrices of the English Roman from the Fell types in possession of the Oxford University Press and which were used as a basis for DTL *Fell. (photo:* DTL *collection)*

Dutch Type Library

DTL DataMaster

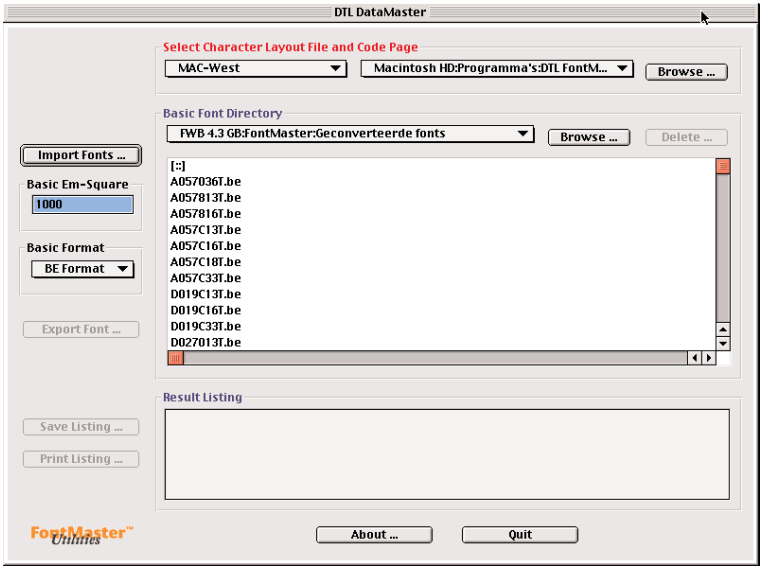
's-Hertogenbosch / Hamburg
Summer 2004



The diagram shows a typical workflow based on the modules of DTL FontMaster.

DTL DataMaster is the module for managing, converting and generating database and font formats. DTL DataMaster converts the PostScript Type 1 and TrueType fonts for Mac OS and Windows to the BE and IK formats that are used by the DTL FontMaster utilities. DTL DataMaster can also generate PostScript Type 1 and TrueType fonts for Mac OS and Windows. In addition, DTL DataMaster creates OpenType fonts according to the Adobe standard. OpenType support in DTL DataMaster is based on Adobe's OpenType SDK and therefore all existing features from Adobes Pro fonts are covered. DTL DataMaster also provides an easy way to generate OTFs also for nonexperts; simply selecting different layouts and/or feature files is sufficient to create fonts with different character and feature sets.

The hints that DTL DataMaster generates for the TrueType fonts can be imported into Microsoft's VTT (Visual TrueType) and used as a basis for delta hinting. The data associated with naming the fonts for the various formats for Mac OS and Windows is saved in special database files, which are platform independent.



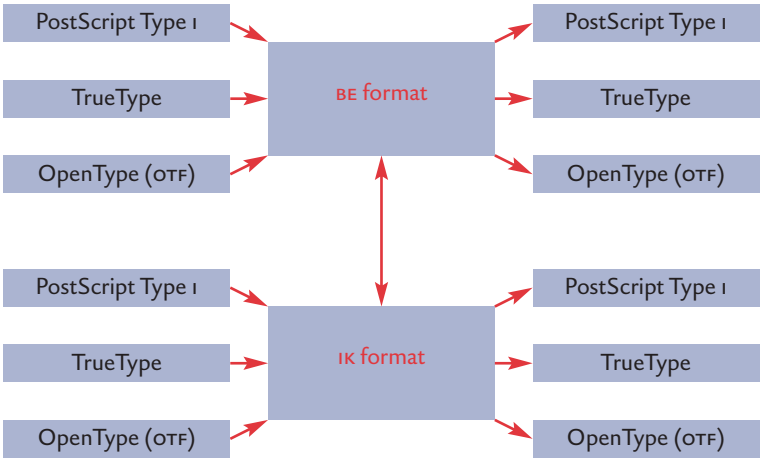
Starting DTL DataMaster

The first dialog has two major functions:
–*Import fonts*
–*Export fonts*
Furthermore you can select a Character Layout File (*.cha) and to change the resolution of the imported font. DTL DataMaster accepts PostScript Type 1, TrueType, OpenType (OTF and TTF), BE and IK format for input. The BE format can be converted to IK format and vice versa. From the BE and IK format it is possible to generate PostScript Type 1, TrueType and OpenType (OTF and TTF) fonts.

OpenType is a rich specification which allows thousands of possible combinations of language lookups and features. Its quite obvious that writing a GUI for the OpenType tables is a huge task. The DTL FontMaster approach is to try to make it easy to generate an OpenType font.
– The OpenType production is based on Adobe's SDK.
– Both OTF and TTF production are supported.
– DTL DataMaster automatically generates as many features as possible.
– Advanced users can create their own set of features.
– No fancy graphic user interface.

For more information, see Appendix VI: OpenType Technology.

The OpenType production is essentially governed by two files:
– The Character Layout File.
– The OpenType Feature File.



DTL DataMaster accepts PostScript Type 1, TrueType, OpenType, BE and IK format for input. The BE format can be converted to IK format and vice versa. From the BE and IK format it is possible to generate PostScript Type 1, TrueType and OpenType fonts.

```
C urwotf.cha
C 20020403
Version 002.000
Starttable
URWNum;urwcOMP;urwcOMP;UNINum;ANNum;QDNum;PSNum;PSName;KernClass;StatClass
999;;;x0020;32;32;32;space;;
101;;;x0041;65;65;65;A;V;
102;;;x0042;66;66;66;B;V;
103;;;x0043;67;67;67;C;V;1
104;;;x0044;68;68;68;D;V;1
105;;;x0045;69;69;69;E;V;1
106;;;x0046;70;70;70;F;V;
107;;;x0047;71;71;71;G;V;
108;;;x0048;72;72;72;H;V;1
109;;;x0049;73;73;73;I;V;1
```

OpenType production is essentially governed by the Character Layout File and the OpenType Feature File. The urwotf.cha is the default Character Layout File for the OpenType production.

1. Import Fonts ...

Before fonts can be imported a series of selections have to be made, such as the selection of the Character Layout File, the Basic Format and, of course, the Basic Font Directory.

*By default the *.cha files are installed in the same directory as the program files.*

1.1 *Character Layout File and Code Page*

The code page defines the layout of the font. A code page consists of 256 character slots, of which a part is used for system functions. There are different code pages for many scripts, such as West-European, East-European, Greek and Cyrillic. There are differences between the layout of

the fonts for Mac OS and Windows; the code pages for West-European are not the same on both platforms. The selection of the code page only has influence on an exported font. It is is of no importance when importing a font into DTL DataMaster. The choice of the Character Layout File (*.cha) is very important when a font is imported. A BE or IK database does not contain any information concerning the PostScript name of a character (like *adieresis* for the ä) or Unicode number (00E4 for the same character).

Based on the original character names and Unicode numbers of the original PostScript Type 1 or TrueType fonts, each character gets a decimal Character Number in the BE or Ikarus database. This process is reversed when a font is generated; each character gets a PostScript name based on its Character Number and in case of TrueType and OpenType fonts, a Unicode number as well. For more info, see the appendices III and IX.

For importing and exporting PostScript Type 1 or TrueType fonts *beeditor.cha* should be selected. For importing and exporting OpenType fonts select the *urwotf.cha* file. An OpenType font produced with the *beeditor.cha* will possibly not support all OTF features because of the simple fact that not all characters will be exported.

For more technical details about the Character Layout Files and Character Numbers, see Appendices III and IX.

1.2 *Basic EM-Square*

The resolution of the description of a font can vary, although there are standards for the different font and database formats. The resolution is defined in the *EM-square* and its units have no absolute size nor any relation to the resolution of any output device. The standard for PostScript fonts is 1000 x 1000 units and for TrueType 2048 x 2048 units. This does not mean that for these font formats no other EM-squares can be used. However the behaviour of such fonts can be unpredictable if the standards are not followed.

The default EM-square for the Ikarus format is 15000 x 15000 units. For the BE format an EM-square of 1000 x 1000 units is recommended. The smallest EM-square supported by DTL DataMaster is 256 x 256 units.

NOTE: The selection of the code page only has influence when a font is exported. However, the choice of the Character Layout File (*.cha) is very important when a font is imported.

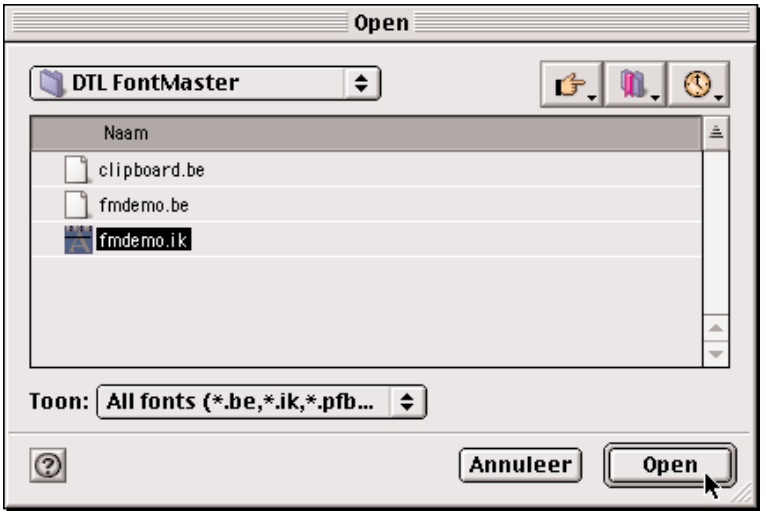
For importing and exporting PostScript Type 1 or TrueType fonts the beeditor.cha file should be selected. For importing and exporting OpenType fonts select the urwotf.cha file.

1.3 *Basic Format*

There are two options: BE format and Ikarus format. The selection normally depends on the editor that will be used: DTL BezierMaster or DTL IkarusMaster. In case DTL DataMaster is used only for converting PostScript Type 1 or TrueType fonts into other formats the default BE format is normally the best choice.

1.4 *Basic Font Directory*

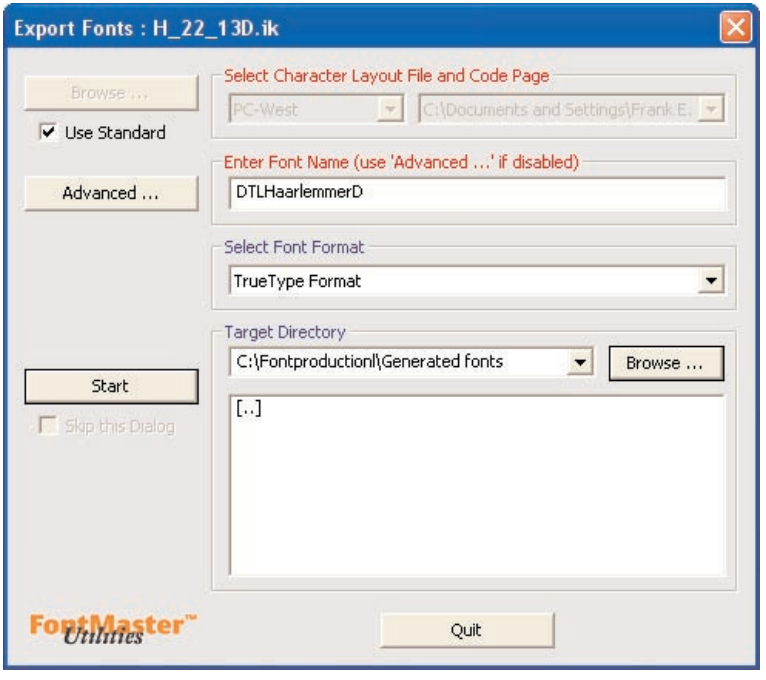
The Basic Font Directory is the place where the converted fonts are stored. So, although named 'Basic' this is the *target* directory. By default this the directory where the FM modules are installed. The *Browse ...* button can be used to select a different directory or folder.



After all selections are made, the *Import Fonts …* function can be selected and the fonts or BE and IK data can be imported from the *Open* dialog.

2. Export Fonts …

For exporting fonts, first make the same selections as for importing fonts. Next the *Export Fonts …* function must be selected for opening the *Export Fonts* dialog.



If a font is imported in DataMaster, all information concerning Font Naming and Copyright Information, Common Metrics Information, Unique PostScript ID, etc. is stored in the UFM file. This file gets the same name as the imported font. In case the file name of the font is changed, the name of the corresponding UFM file must be changed accordingly. Details about the UFM file are revealed in Appendix IV.

At the Dutch Type Library the database names consist of eight characters, like A057016T.be for DTL Argo T Bold (FullName) and F063C14T.ik for DTL Fleischmann ST Medium (FullName). The eight characters guarantee that exchanging the files between (older) platforms does not have any influence on the file names.

2.1 *Character Layout File and Code Page*

Before a font can be generated, the Character Layout File and the Code Page must to be selected. For PostScript Type 1 and TrueType fonts the Character Layout File *beeditor.cha* file is recommended. For the production of OpenType fonts, the *urwotf.cha* file must be selected to preserve the implementation of all the OTF features. The selection of the code page depends on which script the font has to support. Take care that if characters are not available in the font database, the generated font will be incomplete. Use the *Font Administration* tool in BezierMaster or IkarusMaster to check if your font database supports the character set of the code page. Also be aware that if characters are stored in slots that do not correspond with the Character Layout Files, the generated code pages will be incorrect.

Use the Font Administration tool in BezierMaster or IkarusMaster to check the code pages you want to generate with DataMaster.

2.2 *Enter Font Name*

Enter the FontName here. The first time a font database is selected an UFM file will be generated. The FontName is taken from the Font Header and will be shown here. The FontName will be used by the program to generate automatically the FamilyName, FaceName, FullName, FondName, etcetera.

In case you want to control the font naming completely, select the *Advanced ...* button. You have to do this also when the *Enter Font Name* function is disabled. If a UFM file already has been generated, you can not change the Font Name in the *Export Fonts* dialog.

A UFM file contains all information concerning Font Naming and Copyright Information, Common Metrics Information, Unique PostScript ID, etcetera. This file is automatically generated by DataMaster in case a PostScript Type 1 or TrueType font is converted into a BE or IK database or when changes in the *Advanced: view|change UFM and PAR file entries* dialog are saved for the first time. The file is placed in the same directory as the related font database. The UFM file is stored in ASCII format and can be edited directly in a text editor. The UFM file is platform independend; you can move it between Mac OS and Windows. More information about the UFM file format can be found in Appendix IV.

```
Version 002.00E
FamilyName DTLDocumentaST
FontName DTLDocumentaST-Bold
FullName DTL Documenta ST Bold
UniqueID 5060740
Weight Bold
IsFixedPitch false
Ascender 766
Descender -234
UnderlinePosition -133
UnderlineThickness 20
Bodysize 1000
```

Part of a UFM file. The plain text file is stored in the same directory as the font database. The UFM file can be altered directly using a text editor.

The way to handle the naming of the font can differ. The mostly used naming conventions for Mac OS and Windows show differences. On a Mac it is common practice to put all the weight/style names behind the hyphen (FamilyName-MediumItalic), even if this is a medium weight. On a Windows system such naming will not be recognized by the system, except for the Regular and Bold weight. So for Windows the resulting name would be: FamilyNameMedium-Italic. To preserve an optimal exchange of documents between Macintosh and PC with the use of the same fonts, the naming for both platforms should be identical based on the Windows convention, although for this purpose identical FondNames and FullNames should work in most cases.

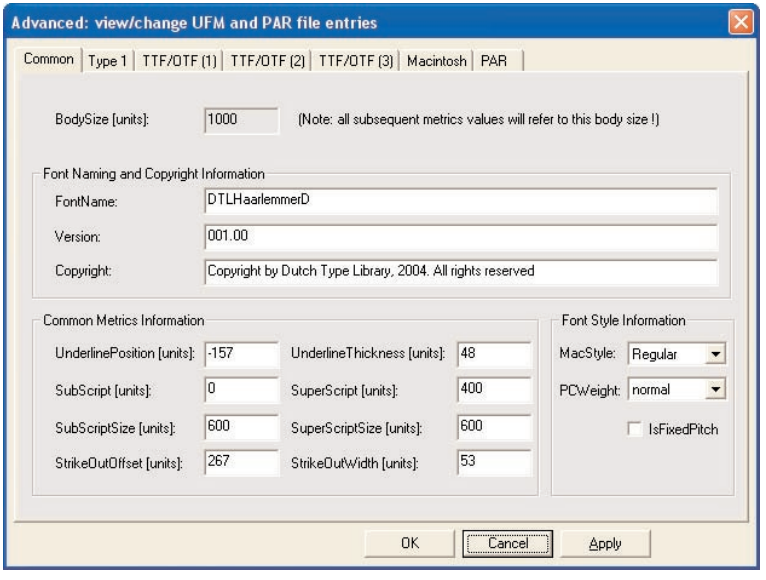
A typical example of font naming, as used by the Dutch Type Library.

For Mac OS:
File: A057014T
FontName: DTLArgoT-Medium
FamilyName: DTLArgoT
Weight: Medium
FondName: DTL Argo T Medium
MacStyle: Regular

For Windows:
File: A057014T
FontName: DTLArgoTMedium
FamilyName: DTLArgoTMedium
Weight: Regular
FaceName: DTLArgoTMedium
FullName: DTL Argo T Medium
PCWeight: Medium

2.2.1 *Advanced ...*

The *Advanced ...* button opens the *Advanced: view/change UFM and PAR file entries* dialog. Here you can control all details concerning for example the Font Naming, Common Metrics Information and Table Information. Be aware of the fact that the behaviour of your font depends on what you enter in the following dialogs:



All details about the items in the UFM dialogs can be found in Appendix IV.

– *Common*

Here you can change the *FontNaming and Copyright Information*. Changing the *FontName* will not change automatically any of the naming in the other UFM dialogs; you will have to do this manually. The *FontName*, also sometimes called the *PostScript Name* must not contain any spaces or special characters, with exception of the hyphen or underscore. It can be a condensation of the *FullName* by removing the spaces. It is customary to

limit its lenght to less than 40 characters.

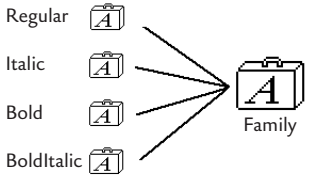
The *Common Metrics Information* is taken from the font that originally was converted in the BE or IK database and in case of a new font it is generated automatically. Any other value can be entered here.

The *MacStyle* in the *Font Style Information* is used on the Macintosh for building font families and is used in the OS 2 tabel for TrueType and OpenType fonts. The *PCWeight* has only an informative function under Windows.

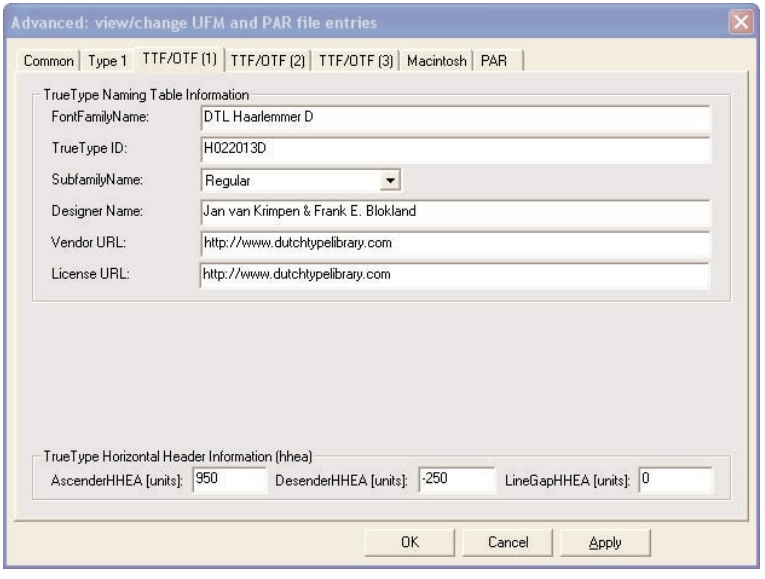
In case you are not going to generate a font family on the Macintosh (an option in the UFM *Macintosh* dialog), always select 'Regular', even if the font is Bold. Otherwise select the appropriate style. The *MacStyle* information does not influence the actual naming but offers the possibility –when a font family is built– to change styles using shortcuts.

The *PCWeight* should represent the weight of the font but in case this is not covered by the nine defined styles ('thin', 'extra light', 'light', 'normal', 'medium', 'semi bold', 'bold', 'extra bold' and 'heavy'), 'Regular' should be selected.

The option *IsFixedPitch* should only be activated for a monospaced font, like Courier.



Although not common practice, on a Macintosh the styles 'Regular', 'Italic', 'Bold' and 'BoldItalic' can be assembled in a font family.



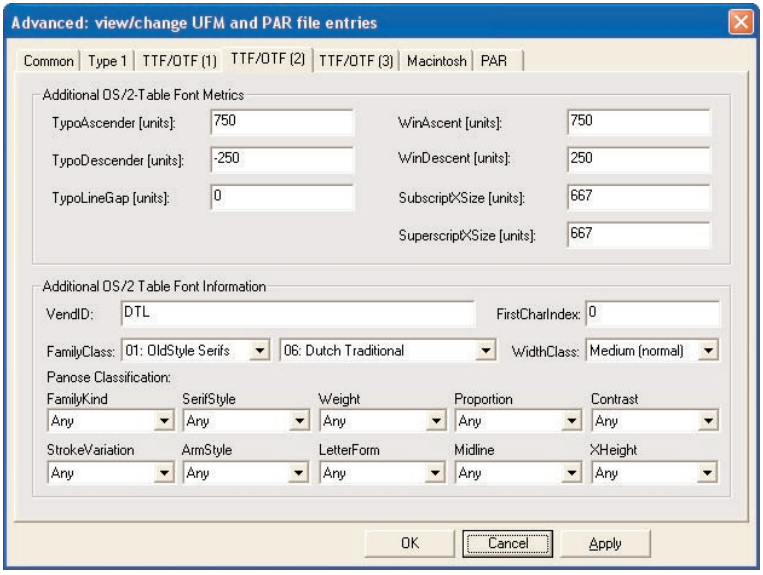
NOTE: For TrueType and OpenType fonts the font style information for the Mac OS is taken from the MacStyle info in the Common dialog.

– TTF/OTF (1)

Here you can enter the *TrueType Naming Table Information*. The *FontFamilyName* is normally identical to the *FaceName* in the UFM *Type 1* dialog. The *TrueType ID* entry should be an unique name and is automatically filled in by the program. At the Dutch Type Library the file

name of the font database is also used as the unique TrueType ID.

The *TrueType Horizontal Header Information* (HHEA) is generated by the program automatically.



The values for the TypoAscender and TypoDescender are by default taken from the information entered in the Font Header in DTL BezierMaster.

– TTF/OTF (2)

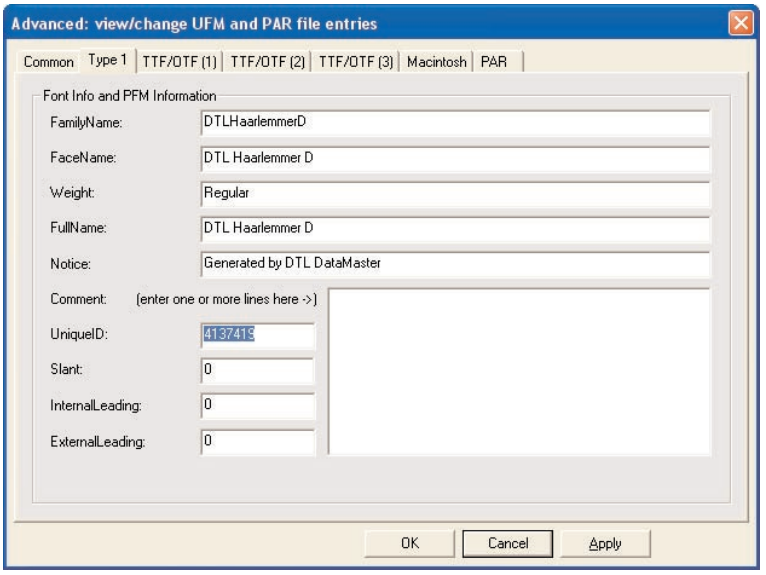
Here you can enter the *Additional OS/2-Table Font Metrics* and the *Additional OS/2-Table Font Information*. The values for the *TypoAscender*, *TypoDescender*, *WinAscent* and *WinDescent* can be defined here. The *TypoAscender* and *TypoDescender* form together the EM-square. The *WinAscent* and *WinDescent* are calculated automatically by the program. Normally the values for *WinAscent* and for *FondAscender* in the *Macintosh* dialog are the same. This is also the case for *WinDescent* and *FondDescent*.

The *TypoLineGap* is by default zero. The *SubscriptxSize* and *SuperscriptxSize* are also automatically calculated by the program.

The *VendID* can be entered in the *Additional OS/2-Table Font Information*. It may consist of four characters at maximum. It is common practice to register this ID via Microsoft, although this is not necessary.

By default the *FirstCharIndex* is set to zero. The default for the *WidthClass* is 'Medium' (normal) and this value should normally not be altered.

The *FamilyClass* and *Panose Classification* don't influence the behaviour of the font but are used to describe the characteristics. This can be of use in case the font has to substitute another font with the same characteristics. The default settings ('No Classification' and 'Any') will of course not support this system but do not harm anything either.



Here the FullName could also be: DTL Haarlemmer D Regular. In that case the FondName in the Macintosh dialog should normally be altered the same way.

– Type 1

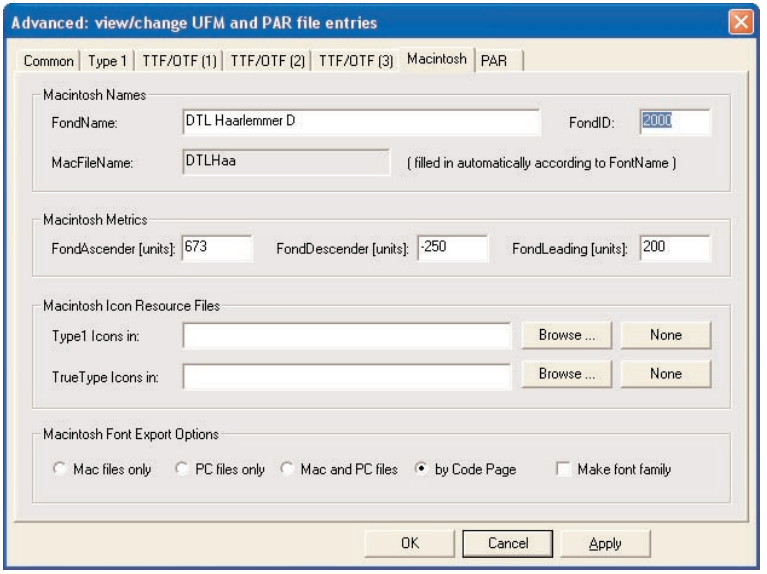
In this dialog most of the naming of the font can be controlled and you can enter the unique PostScript ID here.

The *FaceName* is normally identical to the *FontFamilyName* in the *TTF/OTF (1)* dialog. It may contain spaces. The *FaceName*, also known as *Windows Font Name*, is the name you will see in a Windows program in the font dialog. It is normally the part of the *FontName* before the hyphen; the *FaceName* should not contain style information, like Regular, Italic, Bold or BoldItalic. This means that in case more fonts with different styles are part of a family, the *FaceName* should be indentical in all these fonts. The same is, of course, the fact for the *FamiliyName*. The *FaceName* and *FamilyName* can also be identical, although it is allowed to put spaces in the *FaceName*. Although not common practice, the *FaceName* can differ completely from the *FamilyName*.The *Weight* indicates the boldness of the font. The *FullName* has a commentary purpose. It may contain spaces and looks normally like the *FontName* whereby the name and style parts are separated by a space instead of a hyphen. The *Notice* box can be used to put global information, like a trademark or copyright notice in the font.

In the *Comment* area you can add extra information about for instance the font or the designer.

The *UniqueID* is used to store font information in the cache of a PostScript printer. This way the next time the font is used, it will print faster. It is allowed to fill in just a zero for the *UniqueID* or to enter a number between 4.000.000 and 5.000.000. This range is allocated for free use. For professional font production it is recommended to obtain *UniqueID* numbers from Adobe Systems Inc.

Slant indicates the italic angle counter-clockwise from the vertical. *InternalLeading* and *ExternalLeading* can be used to influence the spacing between the lines in an application. The default is in both cases zero.



Here the FondName could also be: DTL Haarlemmer D Regular. In that case the FullName in the Type 1 dialog should normally be altered the same way.

– *Macintosh*

With this dialog all UFM File information relevant for the use of the font under Mac OS can be controlled.

The *FondName* is the name that will appear in the menu with the font information of an application. It may contain spaces and could be made identical to the *FullName* in the *UFM Type 1* dialog. The *MacFileName* is used by the operating system and it is calculated automatically by DataMaster from the *FontName*. The *MacFileName* has a fixed structure and is constructed as follows: 5-3-3-3-3-3-3… This means that of the first part of the *FontName*, which always has to start with a capital, the first five characters are taken. Of the following parts that start with a capital the first three letters are taken. In case the *FontName* is TestingMyFont-Italic, the *MacFileName* will be TestiMyFonIta.

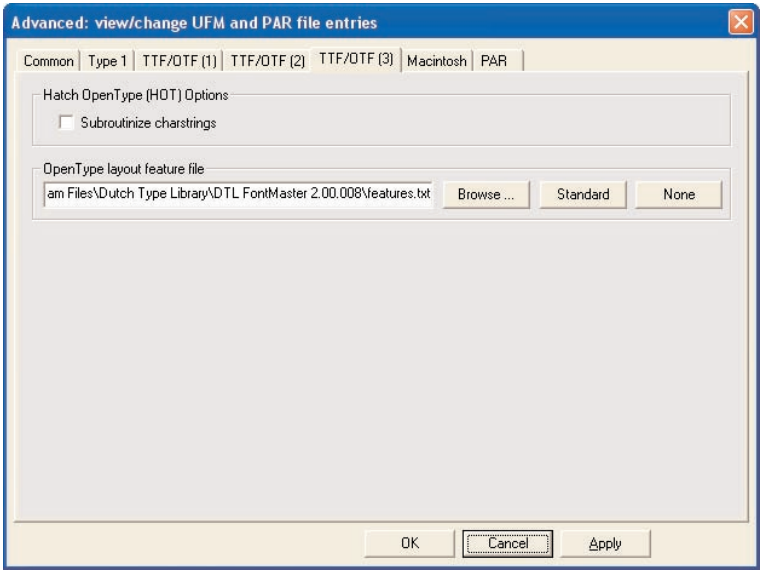
The *FondID* is used by the operating system to identify the font. Each font should have a unique Font Family (Fond) ID.There are ranges for the different scripts. For Roman the range 1024 – 16382 has been defined by Apple for commercial use. Fond ID conflicts should be solved by the operating system automatically. See Appendix IX for more information.

The parts that make the *Macintosh Metrics* information: the *FondAscender*, the *FondDescender* and the *FondLeading* should normally be identical to the

Additional OS/2-Table Font Metrics information (respectively the *TypoAscender*, the *TypeDescender* and the *TypoLineGap*) in the *UFM TrueType (2)* dialog.

The fonts generated with DataMaster will show generic icons. It is possible to select proprietary icons for Type 1 and TrueType fonts from the program by browsing to a directory that contains the *Macintosh Icon Resource Files*. The default icons reside in the directory that contains the DTL FontMaster program files. Be aware of the fact that the icons you select for Type 1 fonts are a somewhat artificial solution; the icons are pasted on the generic icons. The creator information of the font will not be changed. To change the Finder Icons and the BNDL information, etcetera, use DTL IconDropper. You will find details about this program in Appendix XI.

You can use the Macintosh version of DataMaster to generate fonts for Mac OS and PC through the *Macintosh Font Export Options*. This can be done separately but also in batch. Take care of selecting the appropriate code page in the *Export Fonts* dialog. Generating for instance a font for PC with the Mac-West code page selected will result in a non-standard font for the PC. Also batching the production of the fonts for Mac OS and PC will always result in one non-standard font.



More information about the OpenType format can be found in Appendix V.

– TTF/OTF (3)

DataMaster supports the production of OpenType (.OTF) in a very intelligent way. Features are generated automatically on basis of the available characters in the font database. The OpenType support is based on Adobes OpenType SDK. The CFF conversion and the generation of the

GPOS and GSUB tables are integrated. The approach for generating the typographic features is based on the fact that all GSUB features can be generated automatically if the necessary glyphs are in the font. Contextual substitutions are also supported as well as changes to the language system behaviour. There is currently no support by DataMaster for GPOS features, except for kerning.

The default setting for the *Hatch OpenType (HOT) Options* is off.

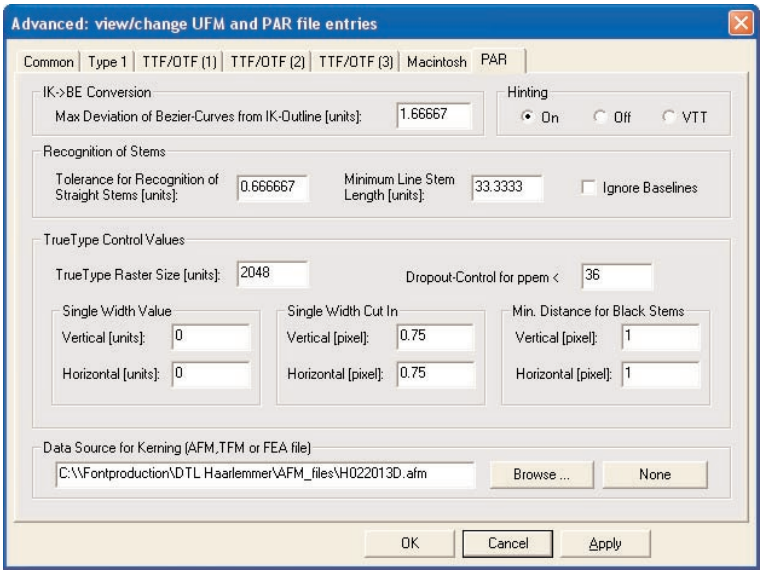
The *OpenType Layout Feature File* is installed under the name *features.txt* in the same directory as the DTL FontMaster program files. The feature file can be edited by an experienced user.

```
# DTL FontMaster Pro Feature File# Version 01.00# -- glyph groupings
@LETTERS_UC = [    A - Z AE Lslash Oslash OE Thorn Eth Aacute Abreve
Acircumflex Adieresis    Agrave Amacron Aogonek Aring Atilde Cacute Ccaron
Ccedilla Dcaron    Dcroat Eacute Ecaron Ecircumflex Edieresis Edotaccent
Egrave Emacron    Eogonek Gbreve Gcommaaccent Iacute Icircumflex Idieresis
Igrave Imacron Iogonek Kcommaaccent Lacute Lcaron    Lcommaaccent Nacute
Ncaron Ncommaaccent Ntilde Oacute Ocircumflex    Odieresis Ograve
Ohungarumlaut Omacron Otilde Racute Rcaron    Rcommaaccent Sacute Scaron
Scedilla Scommaaccent Tcaron Tcommaaccent    Uacute Ucircumflex Udieresis
Ugrave Uhungarumlaut Umacron Uogonek    Uring Yacute Ydieresis Zacute
Zcaron Zdotaccent];@ACCENTS_UC = [    Acute Breve Caron Cedilla Circumflex
commaaccent.cap Dieresis Dotaccent    Grave Hungarumlaut Macron Ogonek
Ring Tilde];@LETTERS_LC = [    a - z ae lslash oslash oe thorn eth aacute
abreve acircumflex adieresis    agrave amacron aogonek aring atilde cacute
ccaron ccedilla dcaron    dcroat eacute ecaron ecircumflex edieresis
edotaccent egrave emacron    eogonek gbreve gcommaaccent iacute
icircumflex idieresis    igrave imacron iogonek kcommaaccent lacute lcaron
lcommaaccent nacute ncaron ncommaaccent ntilde oacute ocircumflex
odieresis ograve ohungarumlaut omacron otilde racute rcaron
rcommaaccent sacute scaron scedilla scommaaccent tcaron tcommaaccent
uacute ucircumflex udieresis ugrave uhungarumlaut umacron uogonek    uring
yacute ydieresis zacute zcaron zdotaccent];@ACCENTS_LC = [    acute breve
```

Part of the DTL FontMaster Pro Feature File that DataMaster uses to generate OpenType fonts. Because the Adobe feature file is used (and also the syntax), everything which is supported by the Adobe SDK is supported by DataMaster too.

In the *Additional OpenType Name Table Entries* information about the designer and relevent URL's can be entered.

A typical example of a GSUB feature.



The values for for instance the Max Deviation of Bezier-Curves from IK-Outline and Recognition of Stems are automatically calculated by the program based on the size of the EM-square.

– *PAR*

In this dialog the values for the parameters necessary for the different conversions can be entered.

With the *IK–BE Conversion* the maximum deviation of Bezier curves from the IK outline can be controlled. Because the standard BodySize of an Ikarus font database is 15000 and the standard for BE is 1000, some scaling is involved with the conversion. The default value for the deviation is 25, assuming an IK BodySize of 15000. Because TrueType fonts are generated always from the BE format, the basic BodySize for all the related actions, like hinting, is 1000 units.

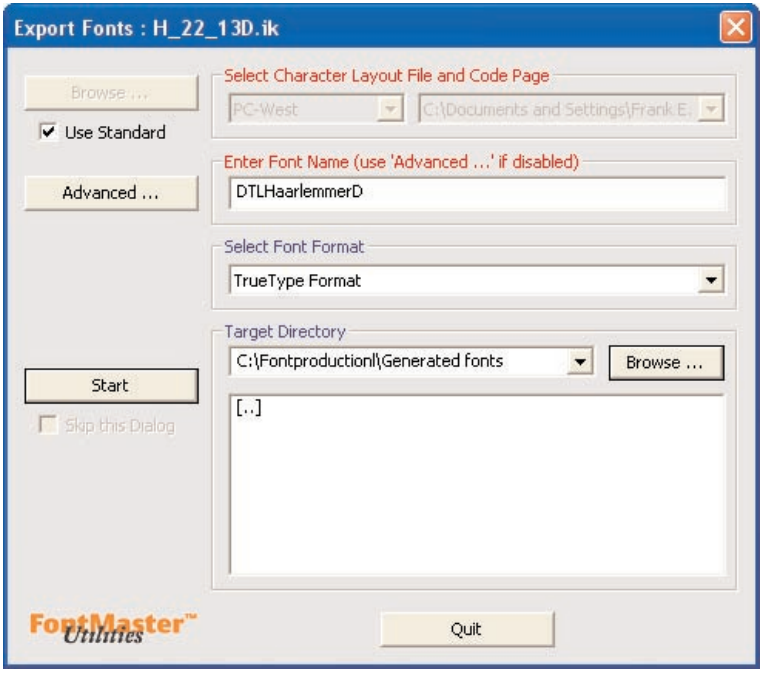
The *Hinting* options are *on*, *off* and *VTT*. The default setting is *on* but in case the of very complex contours, for instance in ornaments, the hinting can be put off. The VTT option makes it possible to generate hinting information that can be imported in VisualTrueType, the de facto program for delta hinting from Microsoft.

The *Recognition of Stems* parameters are for controlling the way stems are handled during hinting by DataMaster. The defined value for the *Tolerance for Recognition of Straight Stems* will make DataMaster handle stems with slight curves that are within this tolerance, as stems with straight lines. The default value is 10, assuming an IK BodySize of 15000. The *Minimum Line Lenght* is the minimal lenght of stems that will contain instructions. The default value is 500, assuming an IK BodySize of 15000.

In case the BodySize deviates from the standard 15000, DataMaster will calculate automatically the values for the *Tolerance for Recognition of Straight Stems* and the *Minimum Line Lenght* based on the default values. The default for the *Ignore Baselines* option is off.

The *TrueType Control Values* are meant for controlling the details of the conversion to TrueType. The default for the *TrueType Raster Size* is 2048; it is recommended not to alter this value. The *Dropout-Control for* PPEM< indicates dat the dropout control is activated for the PPEM's (pixels per EM square) that are smaller than the indicated value. The default is 36. This value can be up to 127 for extremely light weights.

The default for the *Single Width Value* is for both options (*Vertical* and *Horizontal*) zero units. The default value for the *Single Width Cut* is threequarter of a pixel for both directions. The *Min. Distance for Black Stems* has a default value for the *Vertical* and *Horizontal* of one pixel.



TIP: *Produce fonts in batch by activating the Skip this Dialog option.*

2.3 *Select Font Format*

Select the font format you want to generate. Take care of the fact that for PostScript Type 1 and TrueType the Character Layout File *beeditor.cha* should be selected. For OpenType the *urwotf.cha* file must be selected, otherwise the resulting font will not contain all possible OTF features.

2.4 *Target Directory*

Select a target directory for the font that has to be generated.

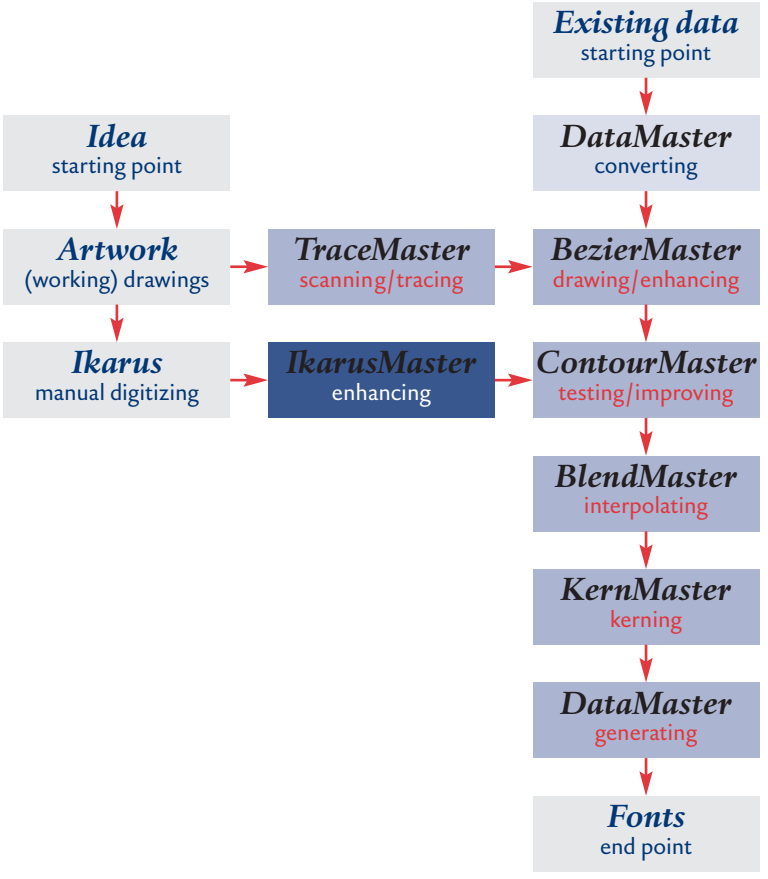
2.4.1 *Start*

Starts the actual generation. Enable *Skip this Dialog* for batch production.

Dutch Type Library

DTL IkarusMaster

's-Hertogenbosch / Hamburg
Summer 2004



The diagram shows a typical workflow based on the modules of DTL *FontMaster.*

DTL IkarusMaster is an editor for the Ikarus format. Almost all of the functionality matches that which DTL BezierMaster offers for the bezier format; there is a Font Administration tool, a Metrics Editor and a number of optimisation functions. The seamless integration with the bezier editor is not only apparent in its similarities with the interface, but also from the fact that Ikarus data can be loaded directly into the background of DTL BezierMaster. Consequently, the conversion of the IK format to the BE format can be fully checked.

DTL IkarusMaster is a fantastic upgrade for the users of the earlier editors of the Ikarus format, including Ikarus M and Ikarus D. The Ikarus format is accepted by DTL DataMaster as input format and this makes it possible to generate PostScript Type 1, TrueType and OpenType fonts directly from the IK data.

User Interface Guidelines

The user interface of the modules of DTL FontMaster follow the standards of Mac OS and Windows.

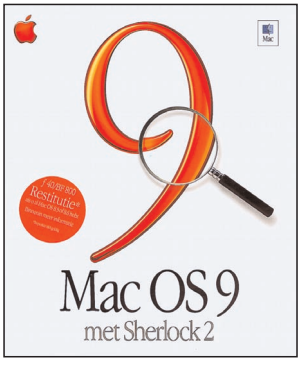
The tool bar showing the functions as graphic symbols can be moved around with the mouse to a desired place.

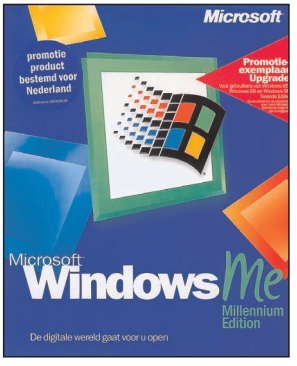
Function selection works as usual: if you click the mouse inside the menu bars, for example **File**, you can select the functions in 'pull down menus' which are displayed on the screen. Select the requested function, for example *Open …*, inside the menu by selecting the item with the mouse. For some functions shortcuts on the keyboard are also available. They are displayed as shown at the right and can be executed without pulling down the **File** menu first.

Under Windows the combination for *New* for example is <Ctrl> + N, on the Macintosh the key is ⌘ + N.

On Windows systems you can also select any function by pressing the <Alt> key and the underlined character shown in the pulldown menu.

A summery of the shortcuts for Mac OS and Windows for DTL IkarusMaster is given at the end of this chapter.

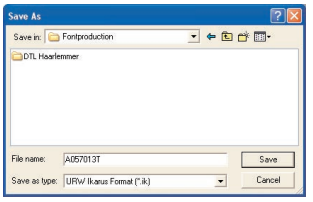




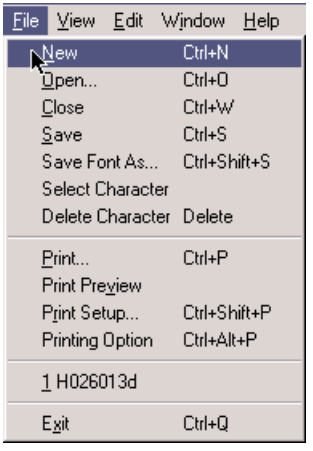
Nomenclature

In this manual as well as in the modules we do not distinguish between glyphs and characters. The term *character* is used many times to indicate an outline description, although a character actually can be represented by different glyphs. For example the character 'a' can look like this:

a*a*·a*a*·a*a*



The screen dumps in this manual were made with alternately the Mac OS and Windows versions of DTL FontMaster.



Shortcuts are displayed in the pull down menus next to the related function.

DTL FontMaster runs under Mac OS 8.6 and higher, including Mac OS X Classic mode and under Windows 95/98, ME, NT, 2000 and XP.

Outline Description

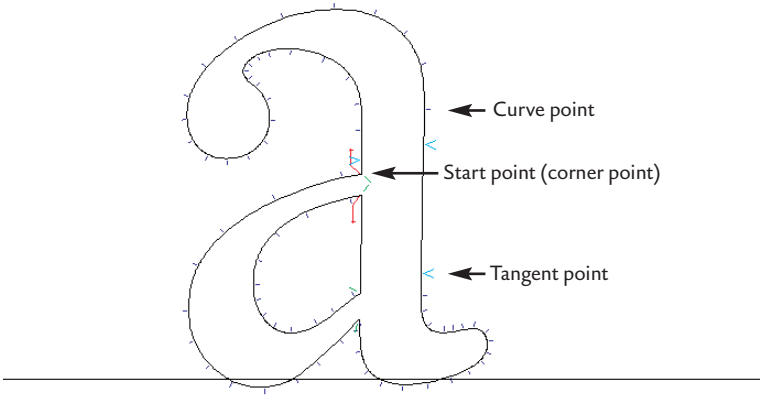
The outlines of the glyphs in the font are described by the Ikarus format. The IK format describes an outline by either straight lines or a Cubic spline section, which is also a third-degree polynomial. Every point of the Ikarus format lies exactly on the contour -in contrast to the Bezier format, where control points are located outside the contour. Therefor the IK format is ideal for working with a digitzing tablet for input.

For a long time the Ikarus format has been the *de facto* standard for creating and manipulating digital fonts, and is still in use worldwide by specialist type design houses. These demanding customers appreciate the high quality of the digital data combined with the system's flexibility, its simplicity and the speed with which fonts can be digitized, edited and converted into other data formats.

NOTE: The IK file format is considered as a database format. Currently one file can contain up to 22000 glyphs. Glyphs are identified by 2-byte numbers (1–65535). The data consists of header, contour header and outline. There is no hinting information in the data. For PostScript Type 1, TrueType and OpenType production additional information is necessary and has to be supplied via text files, like UFM, AFM, etc. (more info about these text files can de found in the appendices).

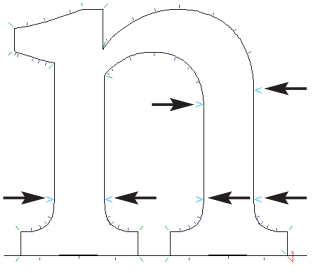
The format specification is public and can be found in Dr. Peter Karow's Schrifttechnologie (Berlin, 1992).

Four different type of point are available to mark the contours:



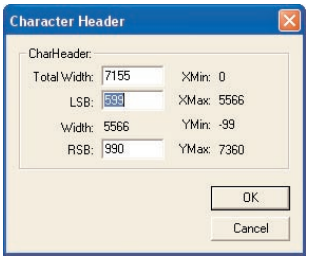
TIP: If you use <Ctrl> + mouse click the program will change the label from Anchor to Smooth Anchor point and vise versa. Using the Change Label function from the Function Tool Bar makes it possible to change straight lines into curves and vice versa, this way adding and removing Control points.

1. **Start points**: only one start point for each contour is allowed. The ideal location for a start point is at a corner. If there are no corner points on the contour, the second best location is at the first (in clockwise order) of two adjacent tangent points. If there are neither corner nor tangent points, the start point should be placed at an extreme curve point. Start points are marked in red by default.
2. **Curve points**: describe curved sections of the contour, by default marked in dark blue.
3. **Corner points**: marking a non-continuous transition between two straight sections, two curve sections or a curve and a straight line.
4. **Tangent points**: These points are marking a continuous tangential transition from a straight line to a curve and vice versa. They are marked in light blue by default.

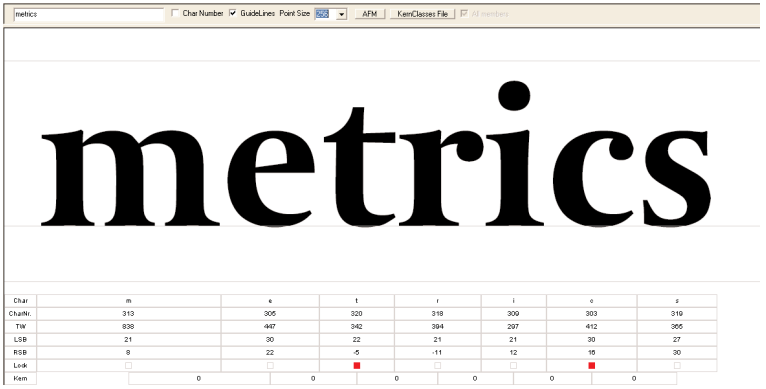


Tangent points are forcing tangent continuity between the adjacent sections.

The available functions of DTL IkarusMaster can operate on three different 'objects':
– Points as described above
– Contours. Contours are consisting of several curves and/or straight lines. They should be closed.
– Characters. These consist of zero, one, or several contours.
There is additional information for the character such as the width and the left and right sidebearings. These values can not be edited interactively in the Character Edit Window, but can be changed numerically in the *Character Header* from the **Edit** menu or in the ***Metrics Window***.



*The character width can be edited with the Change Character Header function from the **Edit** menu.*

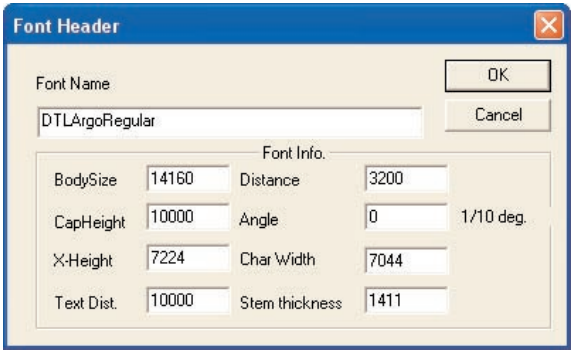


In the Metrics Window the character width can be altered.

The general information about the complete font, such as the Bodysize, is contained in the *Font Header*. For some parameter values it is important to know the bodysize of the font, which usually is 15000 (in standard Ikarus data), 1000 (in a PostScript font) or 2048 (in a TrueType font). Font Header information can be viewed and changed with *Change Font Header* function in the **Edit** menu.

*The Font Header contains general information about the complete font. These settings are used for instance for the V/H Guide Lines function in the **View** menu and for the generation of the UFM file.*

For further information on the Ikarus format please see also Peter Karows books on digital typography, like *Digitale Schriften, Darstellung und formate* (Berlin, 1992).



Selecting points

DTL IkarusMaster uses a standard mouse with one button on the Mac or the left button of the mouse under Windows.

For some functions it is necessary to click a mouse button and at the same time press down the <⇧Shift> key or the <Ctrl> key. On the Mac the command <⌘> key is used.

These combinations will be referred to as <⇧Shift>-mouse button, <Ctrl>-mouse button or <⌘>-mouse button.

The *arrow* tool (–>), also called *pointer* tool, is the default function for selecting objects like points, contours or complete characters. Clicking the mouse button near an outline point will select this point.

The selection will be constrained to a certain radius around the point. This radius can be modified by the user in the **Config menu**. Clicking outside this radius will deselect all selected points.

If no point is selected while a function is executed the whole character will be modified by default.

Arrow or pointer tool

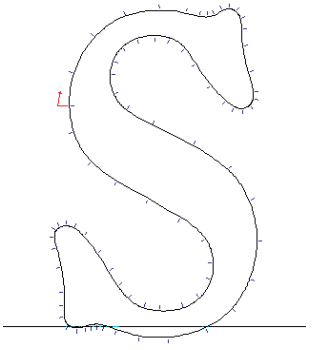
1. Selecting single points

To move or delete for one point you must click near the point on the screen to select it, in other words, select by clicking.

2. Selecting multiple points

2.1 *Collecting points*

Select the first point by clicking on it. While holding down the <⇧Shift> key, click again on another BE point to add it to the selection. You can also deselect an already selected point by clicking on it while holding the <⇧Shift> key.



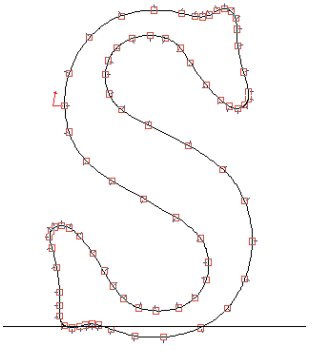
No points selected

2.2 *Window-in Selection*

You can select all points within a user-defined window. To do so, click away from an outline point, hold down the mouse button and drag the cursor over the points to be selected. You can add more points to the already selected collection by pressing the <⇧Shift> key and repeating the window selection. The new points will be added to the already selected ones.

3. Contour Selection

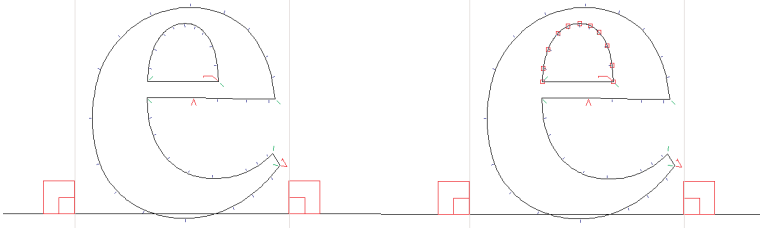
To select a contour, double click near an outline point off the requested contour. To select additional contours click again near an outline point of another contour while holding down the <⇧Shift> key. To deselect a contour click near an outline point off the requested contour while pressing the <⇧Shift> key.



All points selected

4. Character selection

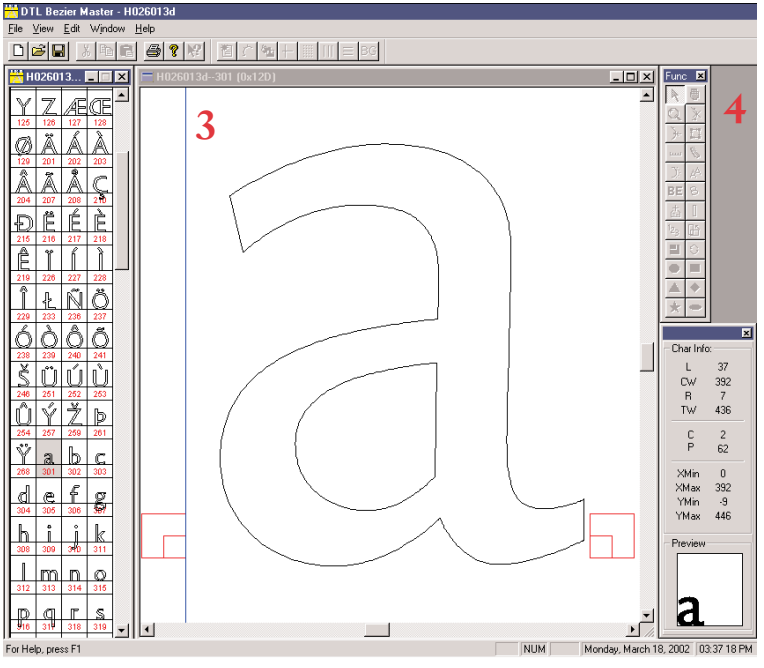
To select all points of a character use the standard shortcut <Ctrl> + A. Some of the more advanced functions, like *Hidden Lines*, assume that all points are to be processed if none is selected.



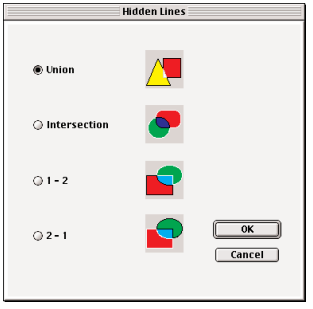
A glyph can also be selected by double clicking while positioning the pointer tool inside the contour.

Screen Layout

The DTL BezierMaster editor has the following components in the screen layout:



1. Tool Bar
2. Character List Window
3. Character Edit Window (several windows are possible)
4. Function Tool Bar
5. Character Display Info and Preview
6. Status Bar



The Hidden Lines function from the ***Special*** *menu.*

Menu functions

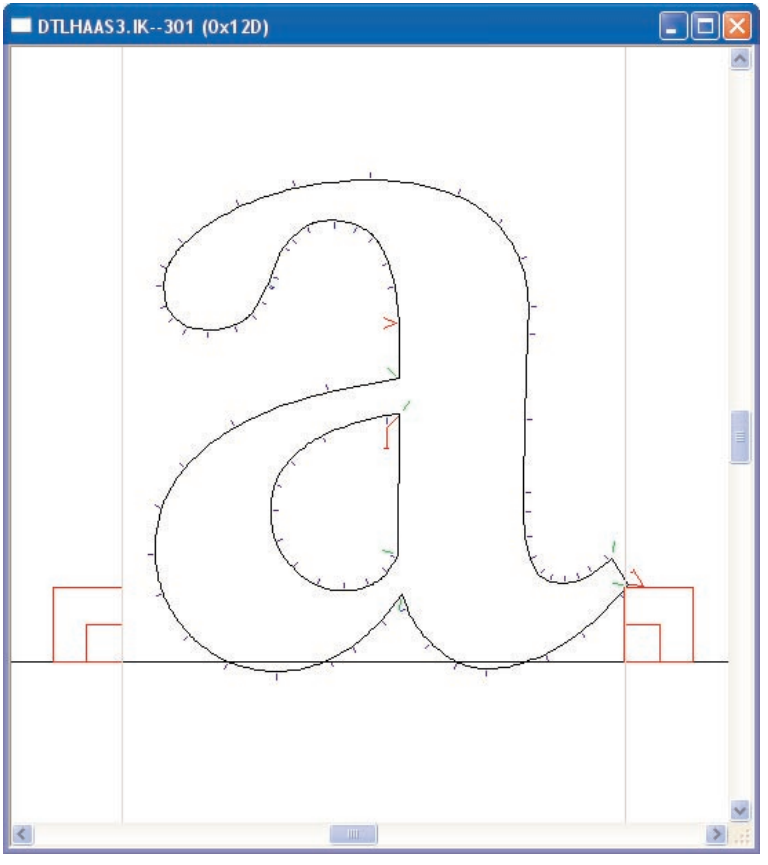
The menu functions are selected from the pulldown menus or with the defined shortcuts from the keyboard.

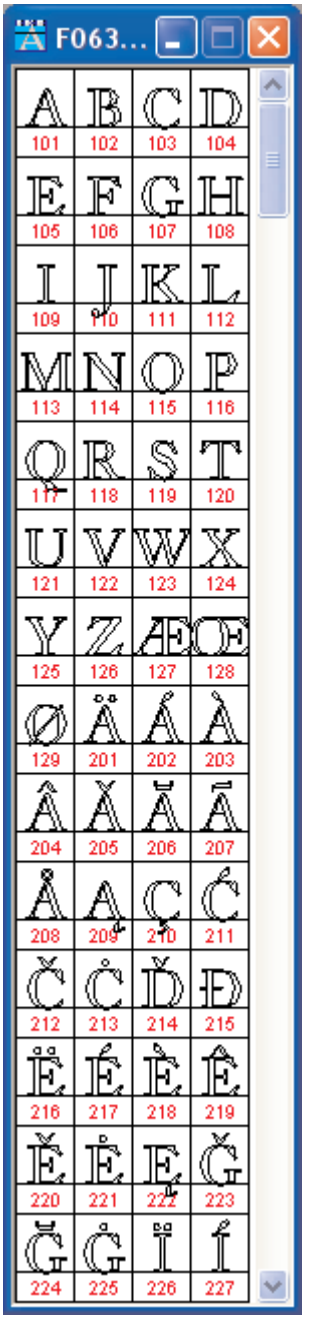
The following menus are available while the Character List Window (right) is active:

File View Edit Window Help

In the Character Edit Window the actual designing takes place.The following menus are available while the Character Edit Window (below) is active:

File Edit View Tools Function Special Config Window Help





The Character List Window (top) shows an overview of all characters in the database. The Character Edit Window is shown left.

FILE MENU

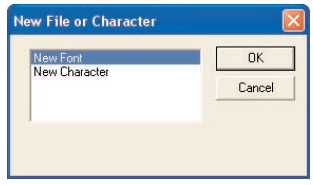
The file menu has the following functions in the pull down menu:

New (⌘ + N) (Ctrl + N)

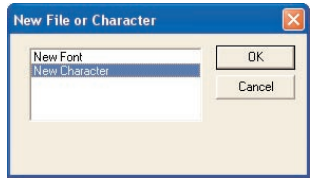
This function allows you to create a new font or a new character in an already open font.

1. New font

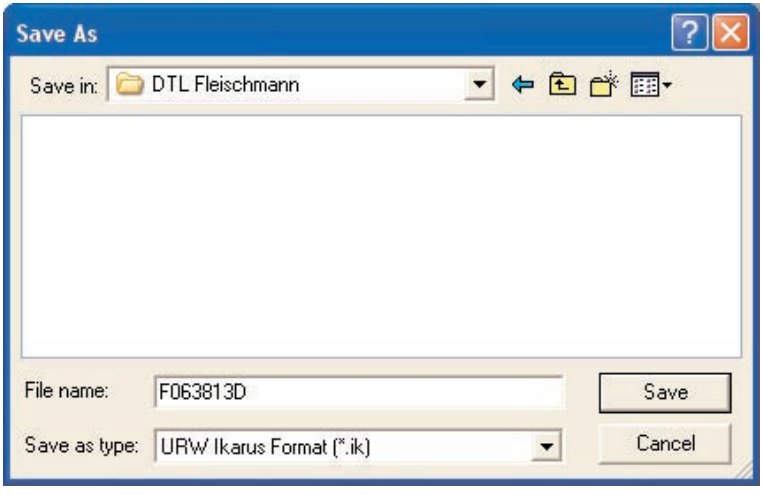
Generates a new IK font. You should name and save the font first. Afterwards you insert the character number of the character in the new font you wish to create.



Before a new IK character can be created, first a new IK font has to be generated.



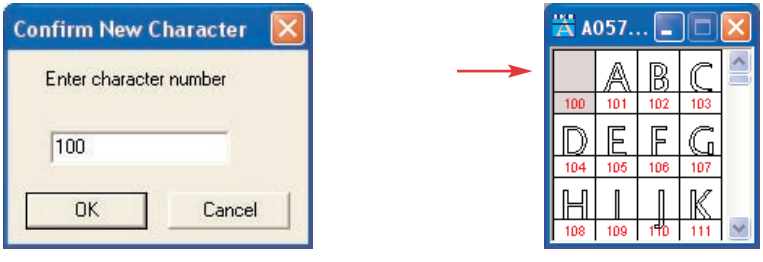
After a new IK font is generated, a new IK character can be created.



2. New Character

Insert the number of the new character you wish to insert into your currently edited font in the following dialog box.

Allowed character numbers are in the 16-bit range i.e. from 1 to 65535. 0 (zero) is not an allowed number. Character numbers can be found in Appendix IX: *Character number listing*.



TIP: Instead of generating a new character by entering the appropiate BE number, the Font Administration tool from the View menu can be used. Double clicking with the mouse on a cell of a particular codepage will open the character and add the IK number to the database accordingly.

After a character number has been defined, a new slot in the Character List Window is generated.

Open (⌘ + o) (Ctrl + o)

Opens an existing font using the standard open file dialog box. After succesfully opening the font the program will display all characters in the Character List Window. This window can be used to select characters to be edited by simply clicking into the small window. Several character edit windows can be opened simultaneously.

On Windows systems there is the option to use abbreviations (archives) for predefined directories. If a certain flag in the Windows registry is set, the program will accept only input from archives and display a special open dialog.

Close (⌘ + w) (Ctrl + w)

This function works differently depending on which window is currently active. If the character list window is active, it will close the font which you are working on. If you have changed some characters, you will be asked whether to save or not save the changes you made. All open character edit windows will be closed too. If the character edit window is active only this particular window will be closed.

Save (⌘ + s) (Ctrl + s)

Saves the data to disk. It is recommend to use this function frequently to avoid a loss of data. This function works both in the Character List Window and the Character Edit Window.

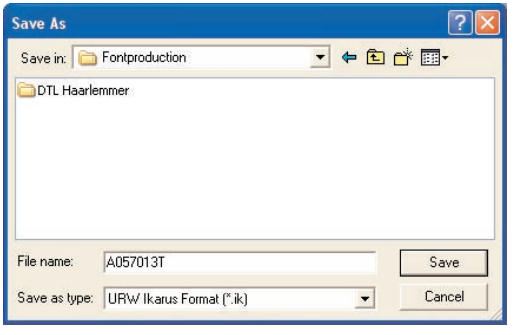
Save as (⌘ + ⇧Shift + s) (Ctrl + ⇧Shift + s)

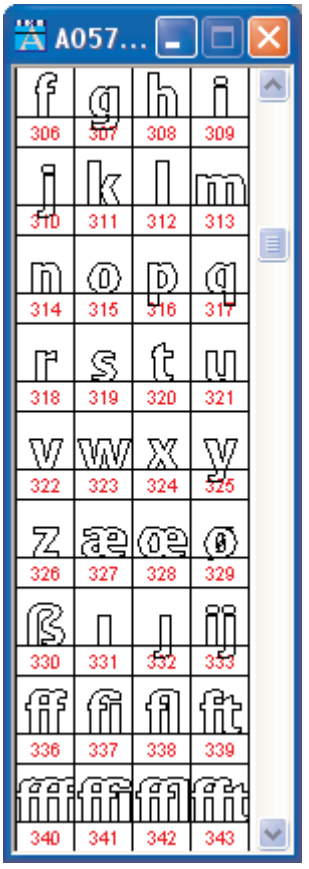
This option functions different depending on the active window:

1. Character Edit Window active

Saves the character you are working on to a different character number in the font database. The Character Edit Window will subsequently contain the new character. The previously edited character will be closed without saving the changes.

This function can be used to duplicate characters to other positions, as a basis to construct new characters, such as accents, from existing ones, or simply to reposition characters.





After succesfully opening the font the program will display all characters in the Character List Window.

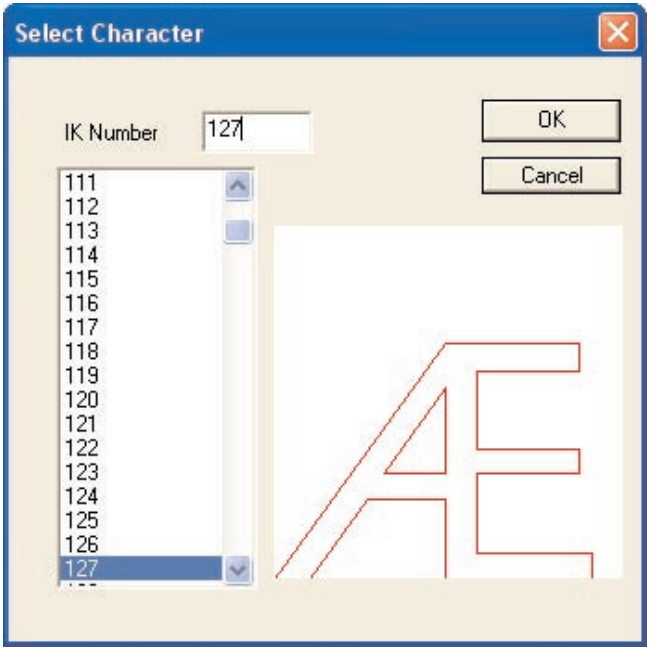
The Character List Window has to be active for saving the font under a different name.

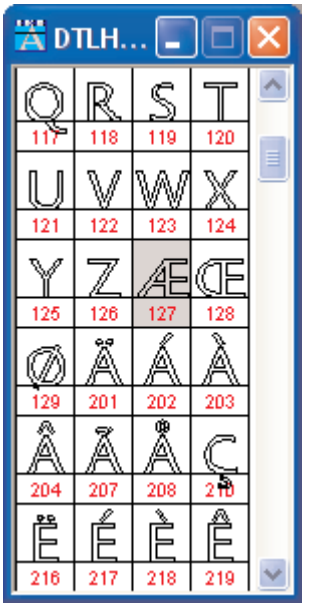
2. Character List Window active

With this function you can save the font under a new name.

Select character

In the following dialog box you can select a character by double clicking its number. The image of the character will be displayed as you click on the number once. You can also enter a character number numerically or use the arrow keys (up and down) to scroll through the character list. Double clicking on a character in the Character List Window gives the same result.





The numbers shown in the Select Character dialog are, of course, the same as in the Character List Window.

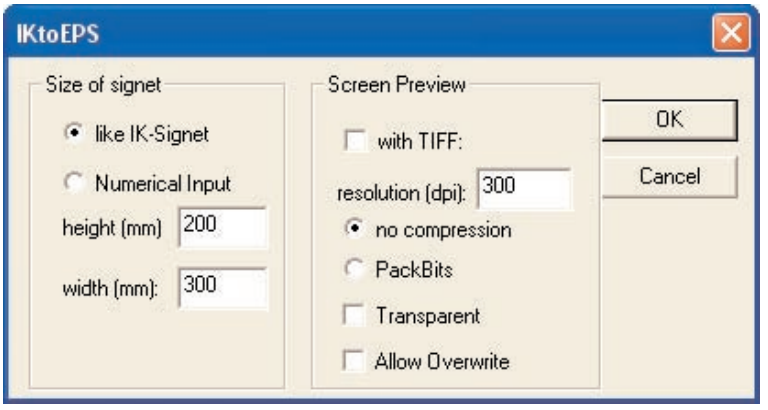
Delete character (← Backspace) (Delete)

To delete a character, just select the character in the Character List Window on the left side by clicking once on the character and then press the <←Backspace> or <Delete> key.

Import EPS file

Reads an EPS file and converts it to IK format. For instance EPS files made with Adobe Illustrator® can be imported. The EPS data will be converted to Ikarus format and merged into the currently active Character Edit Window.

The size and positioning of the IK data is taken from the EPS data and scaled to the bodysize of the font. Please note that the EPS file must have the suffix *.eps* to be recognized. The EPS file must also not contain multiple layers; these have to be removed first before importing the data in the Character Edit Window.



The IK to EPS dialog offers a number of options for the export of EPS files.

EPS Output

Glyphs can be exported individually as EPS data. The same functionality is available for groups of glyphs from the *EPS Output* option in the **Batch** menu.

TIP: To export IK glyphs as EPS data in groups, the EPS Output function in the Batch menu can be used.

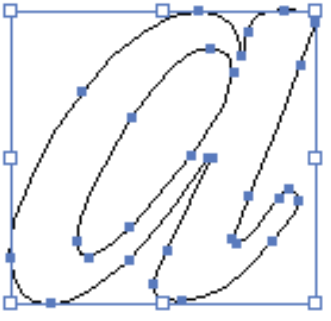
– Size of signet

The glyph(s) can be exported with unchanged size by selecting the option *like IK-Glyph*. This preserves the original relation to the other glyphs in the font database, which makes re-importing (for instance after editing in Adobe Illustrator®) possible without any scaling. With the option *Numerical Input* the bounding box can be scaled to any size measured in millimeters.

– Screen Preview

The EPS files can be provided with a TIFF for screen previewing. As an option the required resolution can be entered. The default reolution is 300 dpi. The TIFF files can be compressed with Apple's *PackBits* format or leaved uncompressed, which is the default setting.

Further options are *Transparant* and *Allow Overwrite*, which let the program overwrite EPS files with the same name. The name of the EPS is taken from the name of the database plus the Character Number. Exporting the glyph with Character Number 302 of the H022013D named database will result in H022013D_00302.EPS. The EPS files are automatically stored in the directory that contains the used font database.



Exported EPS files can be imported for instance in Adobe Illustrator®.

Print (⌘ + P) (Ctrl + P)

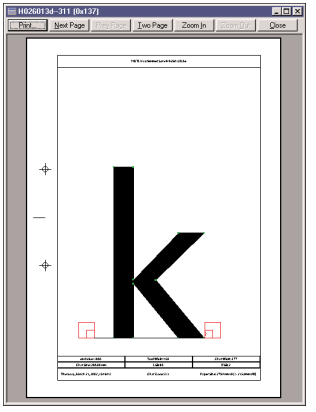
Uses the standard printer driver print dialog. Select the number of pages you wish to print or set the properties of the printer. These depend on the printer you have installed.

Print Setup ... (⌘ + ⇧Shift + P) (Ctrl + ⇧Shift + P)

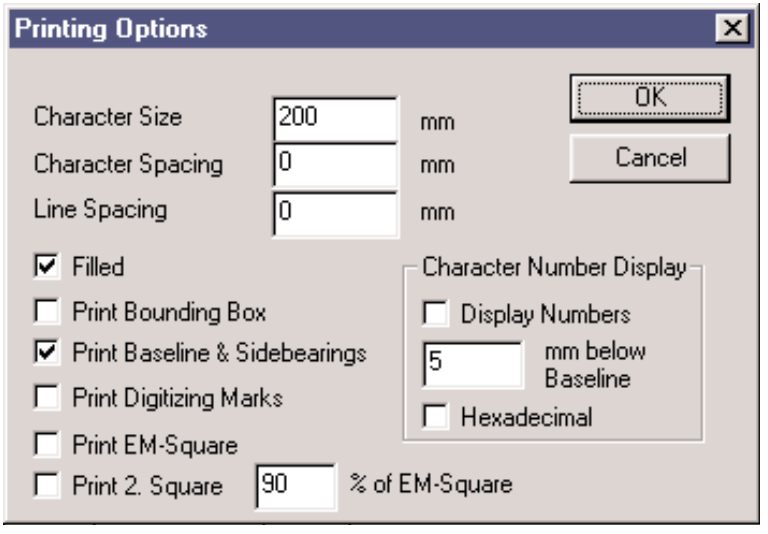
This function allows you to change the printer and printer properties.

Print Preview

This function works differently depending on which window is active.
–If the Character Edit Window is active, a printout of this character will be generated with the selected print options and be displayed in the currently active window as a preview.
–If the Character List Window is active all characters or the selected ones (see **File Menu** –> *Printing Options* –> *Text Options*) will be prepared for printing and the generated page will be shown in the character list window. Since this window is quite narrow it should be enlarged to show the complete page and all the options for the display of the preview.



Print previewing is possible.



A large range of print options is available when the Character Edit Windows is active.

Print Options (⌘ + ⌥Alt + P) (Ctrl + Alt + P)

These options are different depending again on the active window:

1. Character Edit Window active

This function allows you to set different options for proofing of individual characters.

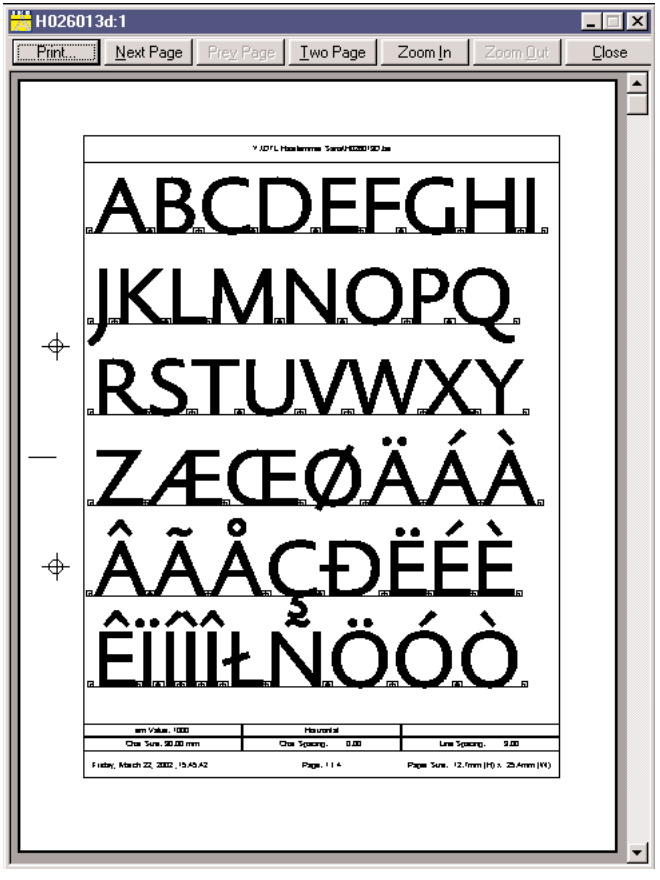
1.1 *Character Size* (bodysize in millimeters).
1.2 *Character Spacing* (spacing in millimeters).
1.3 *Line Spacing* (line spacing in millimeters).
1.4 *Filled* (displays the character solid or as outline).
1.5 *Print Bounding Box* (shows the character bounding box).
1.6 *Print Baseline & Sidebearings* (shows baseline and sidebearings).
1.7 *Print Digitizing Marks* (shows anchor and control points).
1.8 *Print EM-Square* (shows the EM around the character).

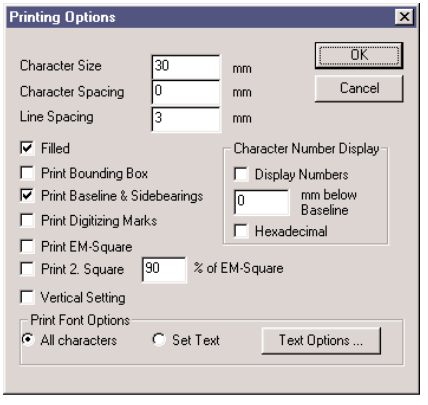
1.9 *Print 2. Square* (shows the facesize for Kanji).
2.0 *Display Numbers* (shows the character number at the set distance below the character).
2.1 *Hexadecimal* (Output of character number in hexadecimal; decimal is the default setting).

2. Character List Window active

This function allows to set different options for proofing of several or all characters. Additional options compared to the printing of a single character are:

2.1 Character spacing (distance between the sidebearings of two characters).
2.2 Line Spacing (distance between the two lines of text).
2.3 Horizontal/vertical setting (option for Kanji setting).

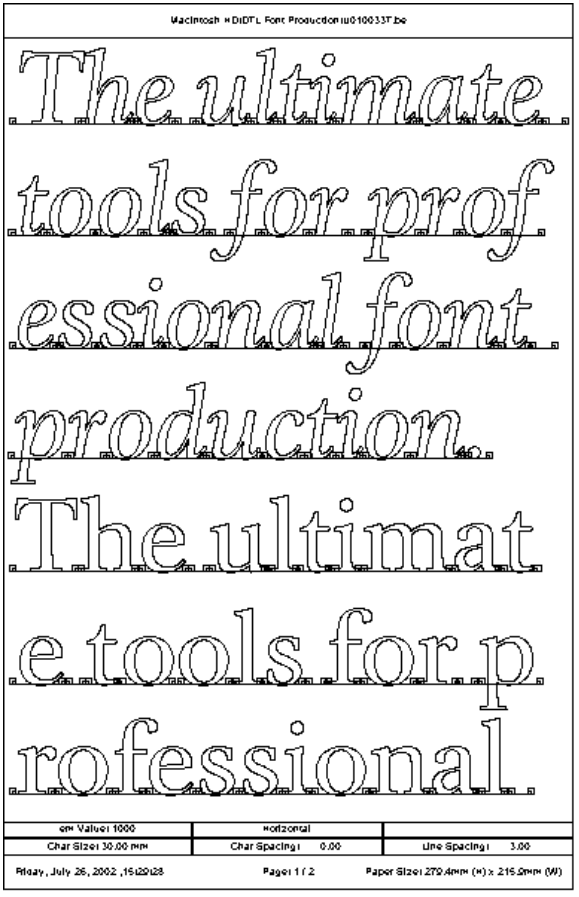


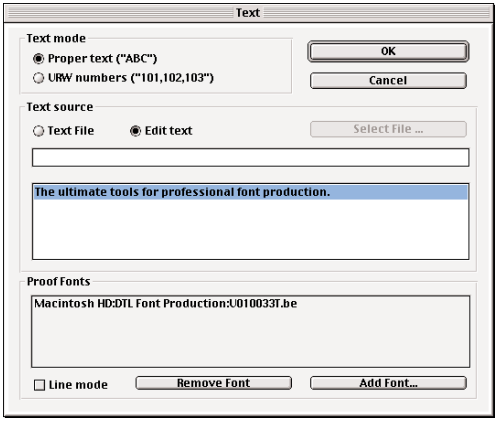


A large range of print options are available when the Character List Window is active. The options selected in the dialog above result in the output shown on the left.

2.4 TextOptions. You will see a dialog which allows the input of character numbers or proof text. For proof text you can select an existing text file or edit text in the Edit Window. You can also choose different fonts for printing the selected text.

After selecting Text Options … in the Printing Options dialog it is possible to enter a text for proofing the font(s), as shown at the left.





(left) In case you have chosen more than one font to display the text, you can display the whole text in the first font, then in the second, the third, etcetera. Note that the line mode will change the Proof fonts after each line of text, not after the complete text.

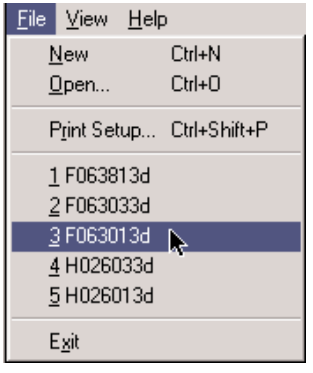
File list, 1 … 2 … 3 … 4 …

Select a font from up to eight font files listed in the **File** menu. These fonts have been opened before and have been memorized by the program automatically.

Exit (⌘ + Q) (Ctrl + Q)

Exit the program. If unsaved data is still in memory, the program will ask you if you would like to save these changes to disk.

On Windows you can also use the standard <Alt> + <F4> to close the program.



The File List, 1 …, 2 …, 3 …, 4 …, function will show up to eight of the most recent used files.

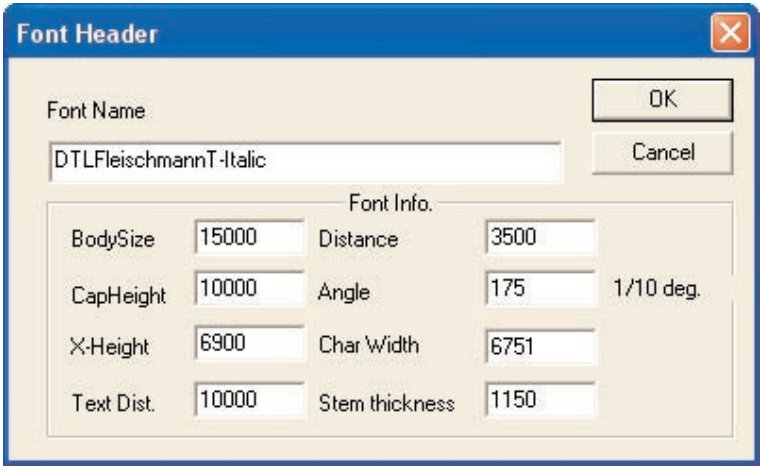
EDIT MENU

These options are different depending on the active window:

1. Character List Window active

Change Font Header (⌘ + ⇧Shift + F) (Ctrl + ⇧Shift + F)

Use this function to numerically change the Font Header as well as to display the content. All values are font specific and might be used for further format conversions.



The font header contains general information about the complete font. These settings are used for instance for the V|H Guide Lines function in the ***View*** *menu and for the generation of the UFM file.*

The *Distance* is the space between the baseline descender line. The BodySize is normally the distance between the extremes of the ascender and the descender.

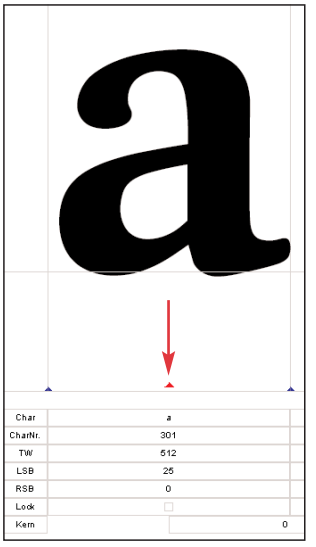
Metrics Editor

This is a very powerful tool to adjust the positions and widths of characters. The Metrics Window can be resized to take full advantage of the screen resolution. The anti-aliased shown Characters can be selected by keystrokes or by character (database) number. The size of the characters can be defined by point size.

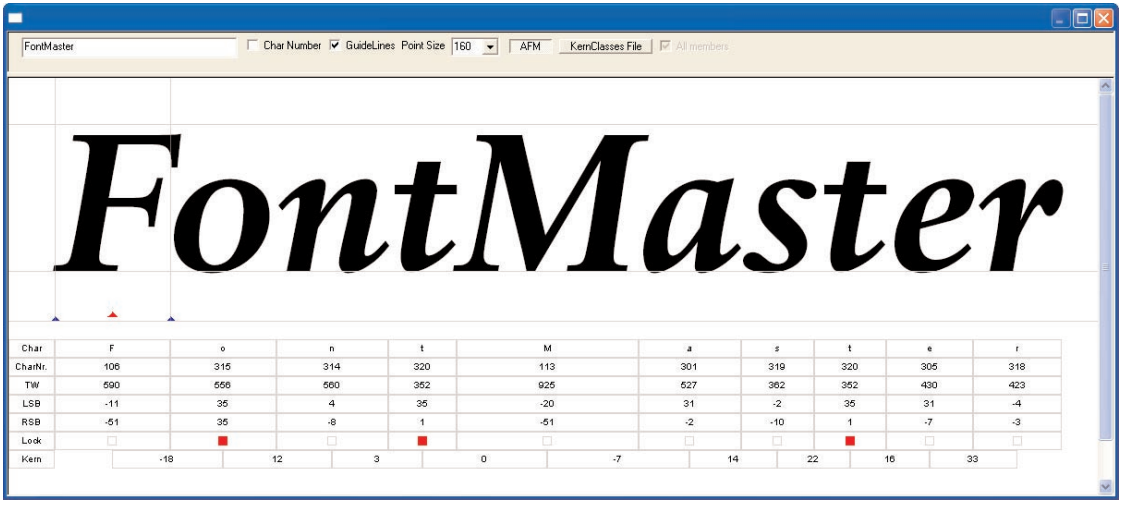
The position of the characters can be changed by selecting and dragging the triangle below the line that marks the *Distance* value in the Font Header (normally the length of the descenders). The triangle marks the centre of the *bounding box*. By default the triangle is blue but in case a character is selected the triangle in the centre is red. Only a selected triangle can be moved. The side bearings can be changed by dragging the blue triangles (these don't change colour) that mark the widths of the characters. Always take note of the fact that moving the side bearings only affects the width of the selected character (marked with the red triangle in the center). The side bearings are shown if the option *guidelines* is selected.

Changes to the width can also be made numerically by altering the values of the side bearings indicated by LSB (left) RSB (right). The changes made in the Metrics Editor are saved automatically to disk. There is of course an undo function. To prevent any errors, there is the possibility to lock the position and the width of the shown characters.

Further options include the import and export from kerning information from AFM file or kern feature file, which for instance can be generated both by DTL KernMaster.

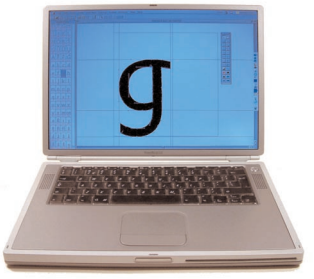


A red triangle in the centre (of the bounding box) marks a selected character.



The Metrics Editor is a powerful tool to adjust the positions and widths of characters.

2. Character Edit Window active



Undo (⌘ + Z) (Ctrl + Z)
Undoes the last editing action. The program supports up to 50 undo levels.

Redo (⌘ + Y) (Ctrl + Y)
Redoes the last editing action with the **Undo** function. Redo supports up to 20 different redo steps.

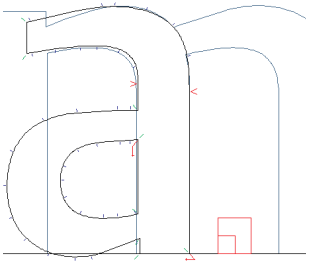
Undo Character Editing
Undoes all changes since the last **Save**.
The program will ask for a confirmation before all edits are discarded.

Cut (⌘ + X) (Ctrl + X)
Deletes and simultaneously copies one or more selected contours to the clipboard.

Copy (⌘ + C) (Ctrl + C)
Copies all selected contours to the clipboard.

Paste (⌘ + V) (Ctrl + V)
Pastes the content of the clipboard into the currently edited character at the position which is defined as the original position plus a fixed offset in X and Y which is currently set to 300,300 for 15000 EM.

Paste (⌘ + ⇧Shift + V) (Ctrl + ⇧Shift + V)
Pastes the content of the clipboard into the currently edited character without offset.



Copying in fore- and background can be done using shortcuts.

Select all points (⌘ + A) (Ctrl + A)
Selects all points of the character.

Copy into Background (⌘ + Ctrl + C) (Ctrl + Alt + C)
Copies the character in the foreground into the background.

Replace by Background (⌘ + Ctrl + R) (Ctrl + Alt + R)
Replaces the character in the foreground by the character in the background.

Paste from Background (⌘ + Ctrl + V) (Ctrl + Alt + R)
Adds the character in the background to the character in the foreground.

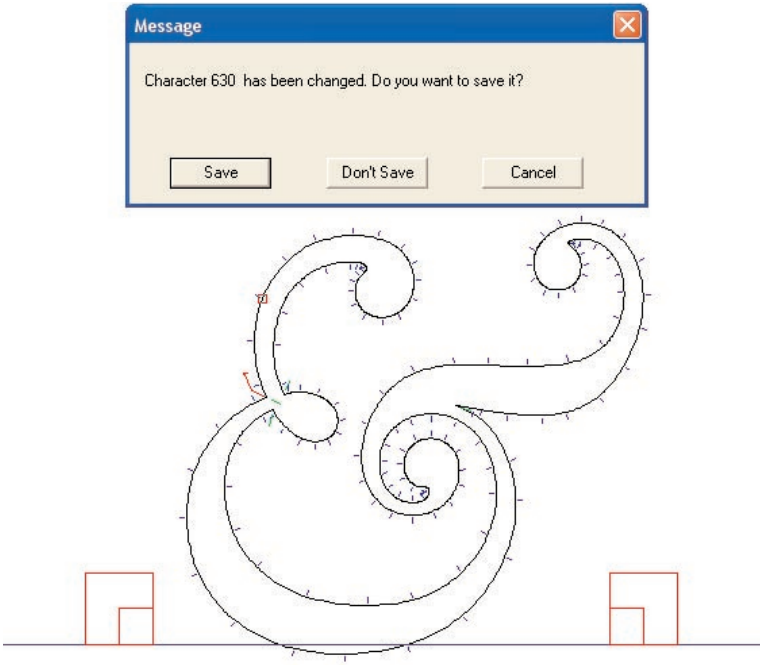
Next Character (⌘ + →KeyRight) (Ctrl + →KeyRight)
Selects the next character in the currently active character edit window

with respect to the character number. If the currently edited character has been modified, the program will ask if you want to save the edits.

If the background is active, the character in the background will be changed too if the same number as used for the foreground character exists.

Previous Character (⌘ + ←KeyLeft) (Ctrl + ←KeyLeft)
Selects the previous character in the currently active Character Edit Window with respect to the character number. If the currently edited character has been modified the program will ask whether to save the edits or not.

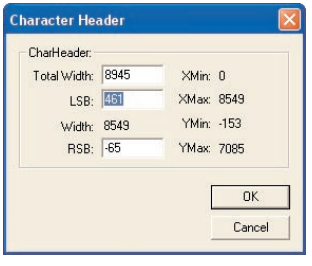
If the background is active, the character in the background will be changed too if the same number as used for the foreground character exists.



If changes have been made to the current character, the program will ask whether to save these or not before showing the next or previous character.

Change Character Header (⌘ + I) (Ctrl + I)
Use this function to numerically change the left side bearing (LSB), right side bearing (RSB) and total width (the width is calculated automatically).

You can edit the fields on the left side. You can not edit the xMin, xMax, yMin and yMax fields. These will be calculated automatically.



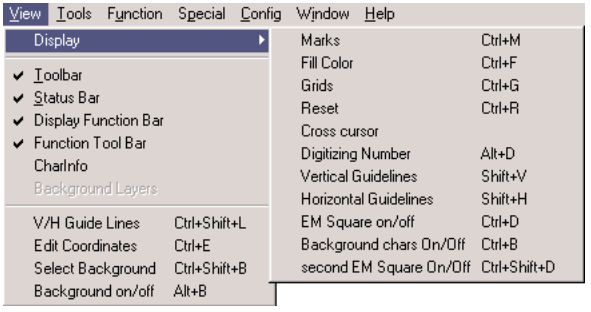
In the Character Header dialog the width and side bearings can be numerically changed.

The side bearings can also be changed by selecting and dragging in the Character Edit Window.

Change Font Header (⌘ + ⇧Shift + F) (Ctrl + ⇧Shift + F)
Use this function to numerically change the Font Header as well as to display the content. All values are font specific and may be used for further format conversions.

VIEW MENU

This menu has different options depending again on the active window. If the Character Edit Window is active you will see the following menu entries.



Display

In this submenu several parameters for the display can be set.

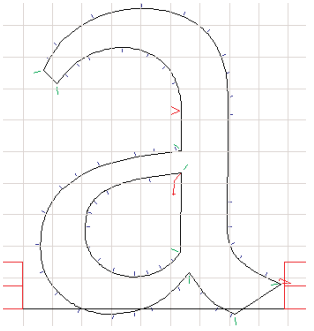
***Display Marks* (⌘ + M) (Ctrl + M)**
Displays the Ikarus points:
– *Start points* (indicated with an arrow)
– *Corner points* (indicated with a long line)
– *Curve Points* (indicated with a short line)
– *Tangent points* (indicated with a v-shaped line)

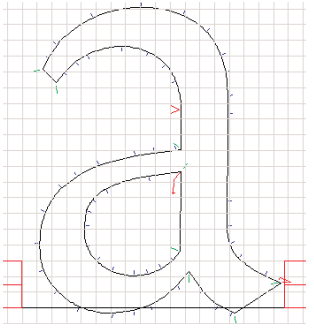
***Fill Color* (⌘ + F) (Ctrl + F)**
Use this function to switch on and off the filled display. The colour is set in the **Config Menu**: ***Editor functions and colors***.

***Grids* (⌘ + G) (Ctrl + G)**
Use this function to switch on and off a grid which is shown in the background. The grid can be defined in the **Config Menu**: ***Editor functions and colors***: ***Grid step***.

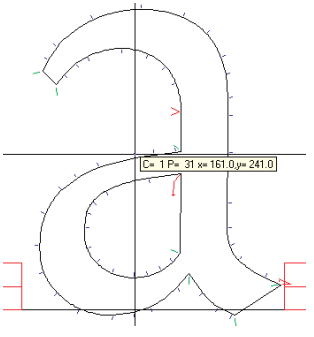
***Reset* (⌘ + R) (Ctrl + R)**
Resets the display to the default size. Use this function also to redisplay the character in case of display problems.

Cross Cursor
Use this function to switch on and off a cross-hair cursor.





The grid is defined in the ***Config Menu: Editor functions and colors: Grid step****.*



The cross cursor is an alternative for the arrow (pointer) tool.

Digitizing Number **(⌥Alt + D) (Alt + D)**
Use this function to switch on and off the digitizing numbers of the anchor and control points, corresponding to the Edit XY list.

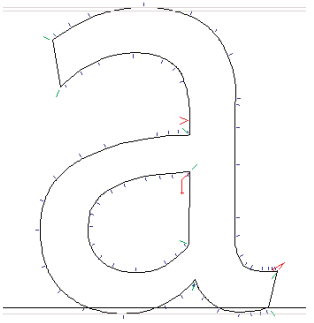
Vertical Guidelines **(⇧Shift + V) (⇧Shift + V)**
Use this function to switch on and off the vertical guidelines as set in **View** –> ***V/H Guidelines***.

Horizontal Guidelines **(⇧Shift + H) (⇧Shift + H)**
Use this function to switch on and off the horizontal guidelines as set in **View** –> ***V/H Guidelines***.

EM Square on/off **(⌘ + D) (Ctrl + D)**
Use this function to switch on and off the EM square.

Background chars on/off **(⌘ + B) (Ctrl + B)**
Use this function to hide and display the characters that are put into the background.

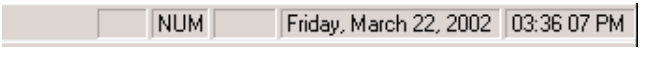
Second EM Square on/off **(⌘ + ⇧Shift + D) (Ctrl + ⇧Shift + D)**
Use this function to switch on and off the second EM square, as defined in the **Config** –> ***Settings*** menu.



*With the **Horizontal Guidelines** switched on, the guidelines set in the **View** menu will be shown.*

Toolbar
Use this function to switch on and off the toolbar with the file and print icon, etc. at the top of the screen. You can use the toolbar to simply select for example, Open or Save, by clicking on the toolbar icon.

Status bar
Use this function to switch on and off the status bar with the date and time at the lower side of the screen.



Display Function bar
Use this function to switch on and off the display functions toolbar at the top of the screen. You can use the display functions toolbar to simply hide or display, for example, marks and grid, by clicking on the toolbar icon.

Function Toolbar

Use this function to switch on and off the Function Toolbar at the right side of the screen. You can use the Function Toolbar to simply select, for example, shift or *Zoom*, by clicking on the toolbar icon .

You can also use the construction tools, such as *Circle* or *Rectangle*, etcetera by clicking on the toolbar icon.

Charinfo

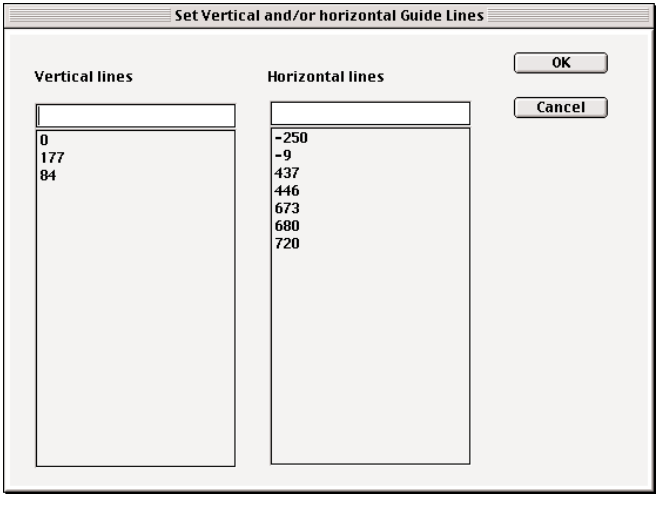
Hides or displays an information window for the characters metrics. The information will be shown on the right side of the screen together with a small filled preview of the character.

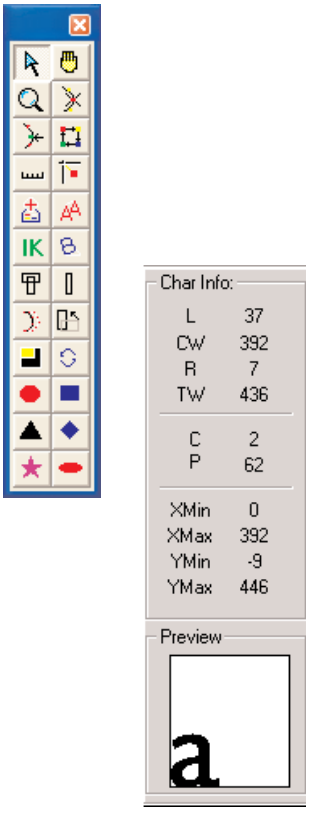
The explanation of the number is as follows:

L	(left side bearing; 1/100 mm)
CW	(character width; 1/100 mm)
R	(right side bearing; 1/100 mm)
TW	(total width; 1/100 mm)
C	(number of contours)
P	(digitizations; number of IK points)
xMin	(position of leftmost IK point)
xMax	(position of rightmost IK point)
yMin	(position of lowest IK point)
yMax	(position of uppermost IK point)

V/H Guide Lines (⌘ + ⇧Shift + L) (Ctrl + ⇧Shift + L)

You can determine vertical and horizontal guidelines in the following dialog. The values for the vertical guidelines start from the left sidebearing (LSB) of the character. After inputting a value, press the <↵Return> key.





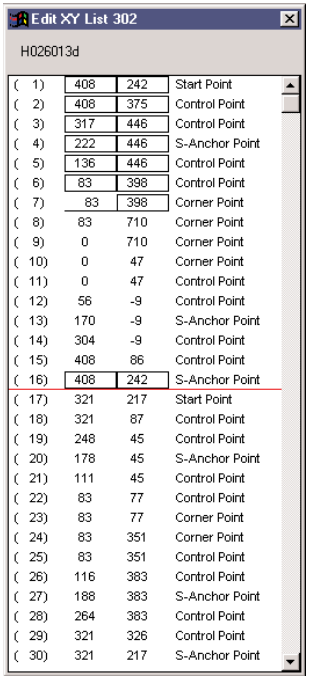
*The functions in the Function Toolbar (top) are described in the chapter about the **Tools** menu.*

*The horizontal guide lines for ascender, x-height and descender are automatically generated based on the values in the Font Header in the **Edit** menu.*

Edit Coordinates (⌘ + E) (Ctrl + E)

Here you can edit the coordinates of all IK points in the following submenu. Selected points are marked in the display as well as in the list.

To shift the selected points numerically just double click on one selected value in the Edit Coordinates window and change it to the desired value by keyboard input. If several points are selected all points will be modified with the same amount. Undo/redo is possible via <Ctrl> + Z/Y.

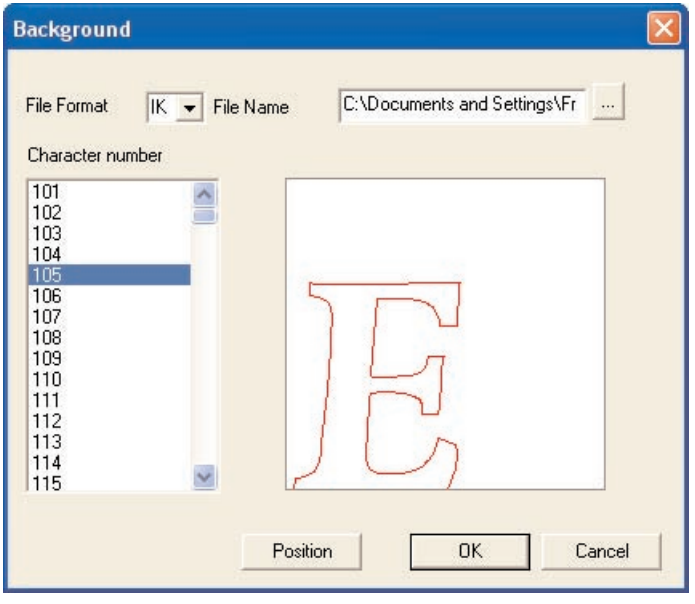


In the Edit Coordinates window the coordinates can be edited nummerically.

Select Background (⌘ + ⇧Shift + B) (Ctrl + ⇧Shift + B)

You can select another IK character from the same font or another font as a background character.

It is also possible to display a scanline character (from scanned input) as background. You can select the background from the dialog shown below.



You can select different formats (IK, II, BE and SC) for the background. It is possible to show IK and SC in the background simultaneausly.

If a background character is selected from a font database that contains more characters, typing ⌘ + → or ⌘ + ← will show not only the next or previous character in the foreground but also the next or previous character from the background font. Closing the Character Edit Window will remove the link to the background font.

Background on/off (Ctrl + B) (Alt + B)

With this function you enable or disable the background. You can also determine whether the outline font in the background is shown with or without marks or modify the position of the glyphs.

With Show background you can make the background visible.

If the Character List Window is active the **View** menu will only display the options *Toolbar*, *Status Bar* and *Font Administration*.

The *Toolbar* and *Status Bar* functions have been described in the previous pages.

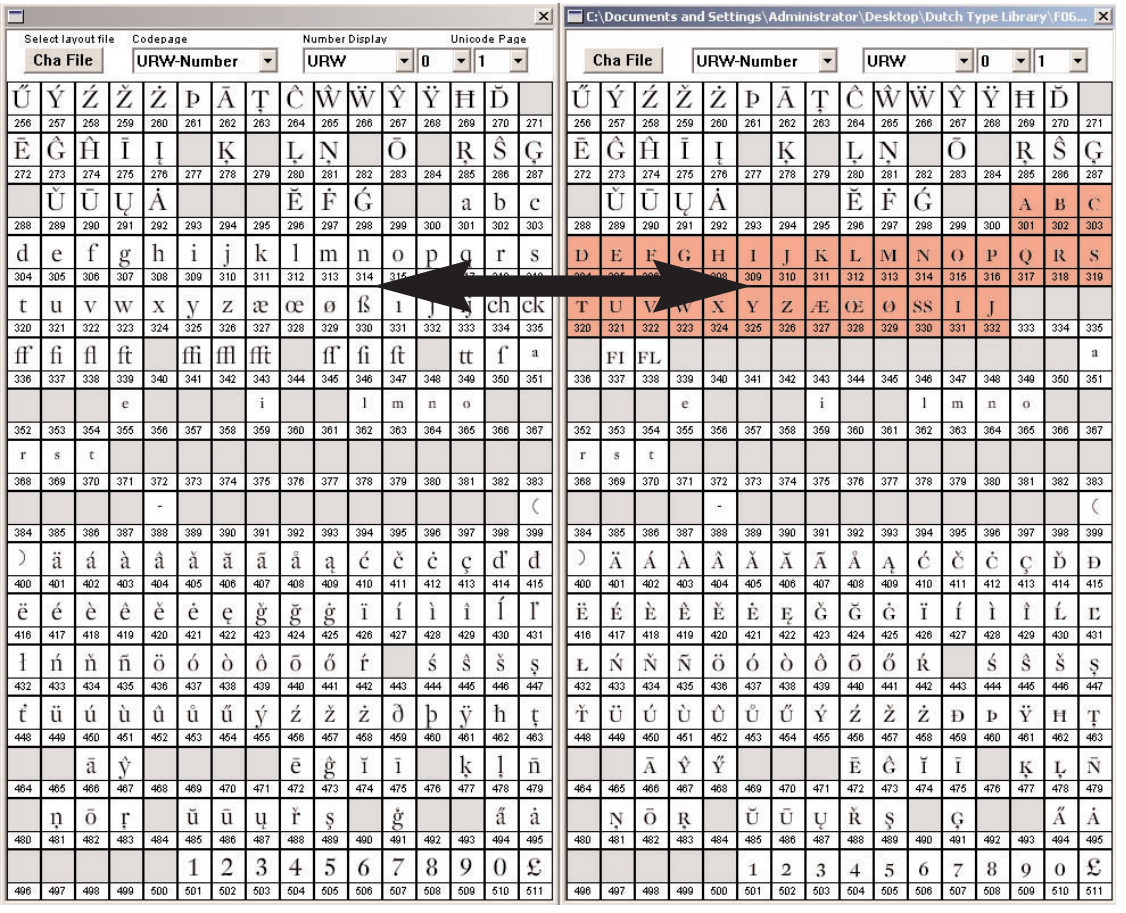
Font Administration (⌘ + U) (Ctrl + U)

This is a very powerful tool for handling and organizing the font database. Because allowed numbers are in the 16-bit range, the database can consist of more than 65.000 characters. In the Font Adminstration window the different code pages that are supported by the character set in the database can be shown beside the Unicode and the Character Numbers. It is possible to copy and paste between the codepages. Pasted characters will be placed in the database automatically under the appropriate number. Newly (re)placed characters are saved automatically. Be aware that undo is not available here!

TIP: Because changes made to the database with the Administration Tool are irreversible, it is recommended to make a back up of the font first.

The current beeditor.cha file supports the following codepages:
–Unicode
–URW-Number
–PC-West
–Mac-West
–PC -East
–Mac -East
–PC-Greek
–Mac-Greek
–PC-Turkish
–Mac-Turkish
–PC-Cyrillic
–Mac-Cyrillic
–PC-Hebrew
–Mac-Hebrew
–PC-Baltic
–Mac-Romanian
–PC-Symbol
–Mac-Symbol
–PC-Kazakh
–Mac-Kazakh

Although characters can be copied and pasted between different IK databases in the Character Edit Window, this functionality is limited to one character at the time. With the Font Administration tool it is possible to copy and exchange (large) ranges of characters between different databases. To select more than one character, hold down the <⇧Shift> key while you select single characters or several series of characters. Characters can be selected in serie by holding the mouse button down and simply dragging the mouse.



With the Font Administration tool characters can be exchanged between different font databases.

The Character Edit Window can also be opened from the Font Administration tool by double clicking on a character.

Different Character Layout Files (*.cha) can be selected. These Character Layout Files are also used by DTL DataMaster when fonts are generated. After installing DTL FontMaster four Character Layout Files are installed in the same directory as the FM modules:

–beeditor.cha

This is the default Character Layout File for DTL BezierMaster.

–TTBAS.CHA

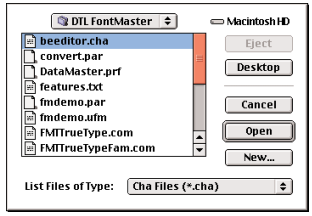
Basically you can ignore this Character Layout File here because normally it will only be used in DTL DataMaster to import fonts.

–urwotf.cha

In DTL DataMaster this Character Layout File must be selected when you want to generate OpenType. All possible OpenType features will only be generated if this .cha file is selected.

–winuni.cha

Basically you can ignore this Character Layout File here because normally it will only be used in DTL DataMaster to import fonts.



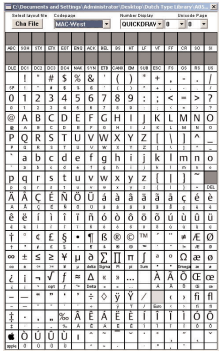
The Character Layout Files are placed in the same directory as the FM modules, libraries and the other stuff.

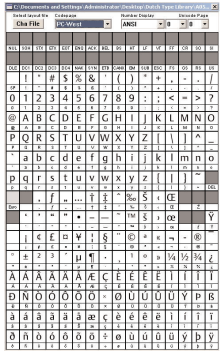
Currently, the beeditor.cha file supports 21 code pages. The Character Layout Files are fully editable and more code pages can be added by the user (see Appendix III).

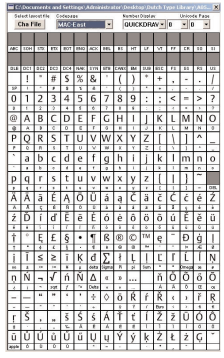
The Number Display shows five options that can be used in combination with the codepages:

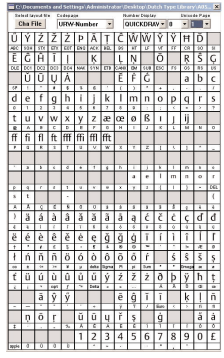
–HEX (hexadecimal notition; default for Unicode)
–URW (URW database numbers)
–ANSI (default for PC codepages)
–QUICKDRAW (default for Macintosh codepages)
–DECIMAL (decimal notition)

Although there are some default combinations of codepages and number displays, each possible combination can be made.







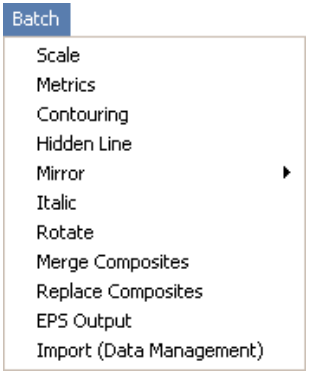


From left to right the code pages for Mac-West, PC-West, Mac-East, and URW-Number.

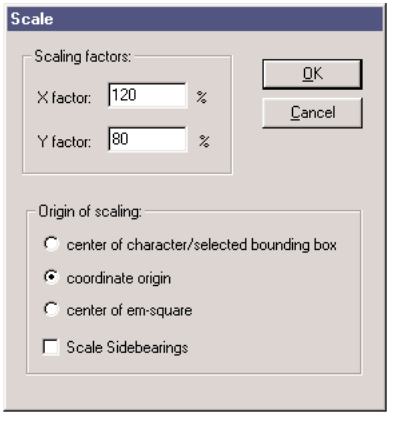
BATCH MENU

This menu is only available in case the option *Font Administration* is selected from the **View** menu. The functions in the **Batch** menu can also be found in the **Tools** and **Special** menus when the Character Edit Window is active. The big difference between the functionality in the **Batch** menu and the other two menus is that in the *Font Administration* tool several or even all characters can be selected for processing while using the same functions in the Character Edit Window only influences the selected character. If no glyph is selected the batch operation will be applied to all glyphs in the database.

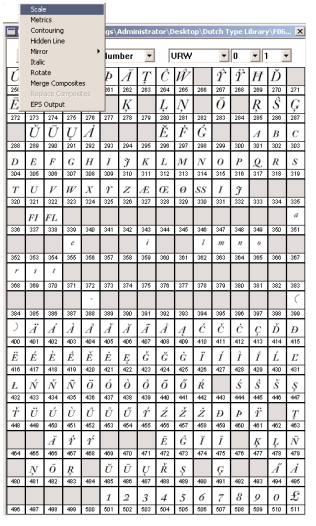
Please pay attention to the fact that changes are irreversible; the operations applied to the glyphs are automatically saved on the disk.



*The options in the **Batch** menu are only available in case the Font Administration tool is active.*



In case no glyph is selected the batch operation will be applied to all glyphs in the database.



Scale

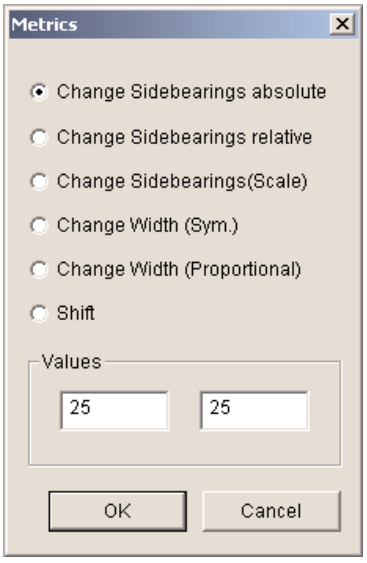
After having selected contours or the whole character, input the scaling factors in X and Y direction in %. You can also determine the origin of the scaling and whether or not you will scale the sidebearings simultaneously.

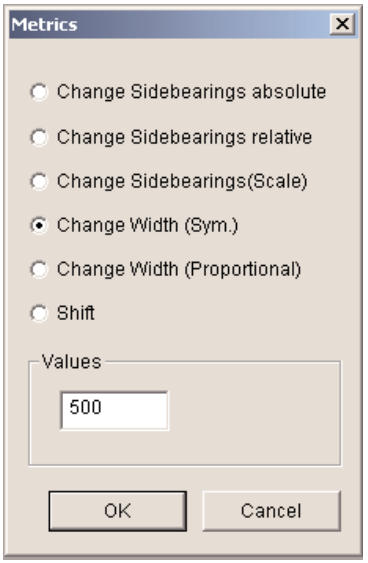
Metrics

There are several options to modify the metrics by changing the widths or sidebearings of the selected glyphs. In the Metrics dialog positiv or negative values can be entered in the input field(s) called *Values*.

– Change Sidebearings absolute

The sidebearings of all the selected glyphs will get the values specified in the two input fields for the left and right sidebearing.





For changing the sidebearing(s) the dialog will show two input fields ('Values'): one for the left and one for right sidebearing. For changing the width(s) there is, of course, only one input field available.

– Change Sidebearings relative

The sidebearings of all the selected glyphs will be modified by the values specified in the two input fields (for the left and right sidebearing).

– Change Sidebearings (Scale)

The sidebearings of all the selected glyphs will be scaled according to the values specified in the input field.

– Change Width (Sym.)

The total width of all the selected glyphs will be made equal to the value in the input field. The glyphs will be centered in the specified width, this way making the left and the right sidebearing equal (symmetrical).

– Change Width (Proportional)

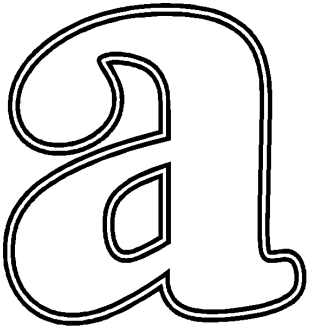
The total width of all the selected glyphs will be made equal to the value in the input field. The left and the right sidebearings will be changed proportionally according to the original relation between the sidebearings.

– Shift

The glyphs will be shifted in horizontal or vertical direction by the specified amount.

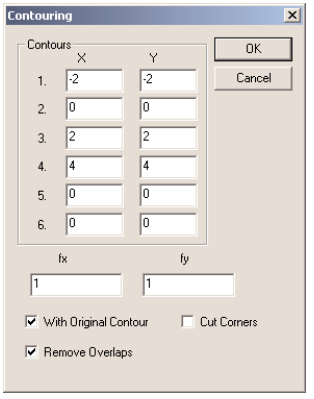
Contouring

This function is used to create additional contours automatically, so called outlined characters. It can also be used to bolden a typeface or make it



An example of contouring.

thinner. Input up to six values in millimetres into the Contouring dialog to create up to the same amount of additional contours. A positive value of for example 2 adds a contour with two millimetres distance from the original contour to the outside. A negative value works to the inside.



Up two six contours can be generated using the Contouring function from the ***Batch*** *menu.*

– Contours

Here you can enter positive or negative values for the contouring.

– fx and fy

The creation of contours can be combined with scaling. The factors specified for scaling in horizontal and vertical directions will be applied on the selected glyph(s).

– With Original Contours

In case this option is selected the original contour will be preserved. This only works if the specified values are positive.

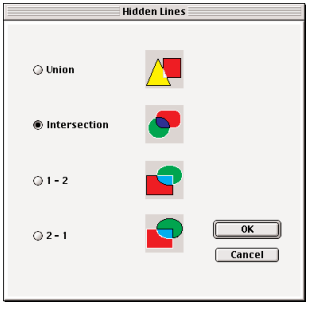
– Cut Corners

Selecting this option will preserve line thickness in the newly generated contour.

– Remove Overlaps

The functionality of this option is comparable with the *Union* function from the *Hidden Line* option and prevents overlapping contours.

The Hidden Line dialog.



Hidden Line

This function merges overlapping contours. It currently works always on the complete character.

– Union

Merges the contours.

– Intersection

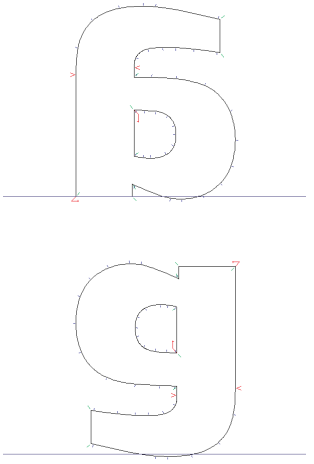
Creates the intersection. Only the overlapping parts remain. The rest of the contours are deleted.

– 1-2

Deletes the second contour and the part of the first contour that was overlapped.

– 2-1

Deletes the first contour and the part of the second contour that was overlapped.



The a on top is mirrored Left <–> Right and the other Top <–> Bottom.

Mirror

This function mirrors the selected contours or the whole character. There are two options.

Left <–> Right

This function mirrors the selected contours or the whole character horizontally around the center of the selected parts.

Top <–> Bottom

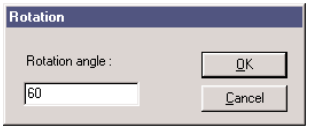
This function mirrors the selected contours or the whole character vertically around the center of the selected parts.

Italic

Use this function to oblique the character electronically. A special selection mode is not required. In this function the character mode is always used. After selecting the function a pop-up menu appears. An angle between –45 and +45 degrees is recommended. A positive angle obliques clockwise.

Rotate

The rotate function works for selected contours or the whole character. Input a Rotation angle; a positive angle rotates clockwise.



The Rotate dialog in which the angle can be specified.

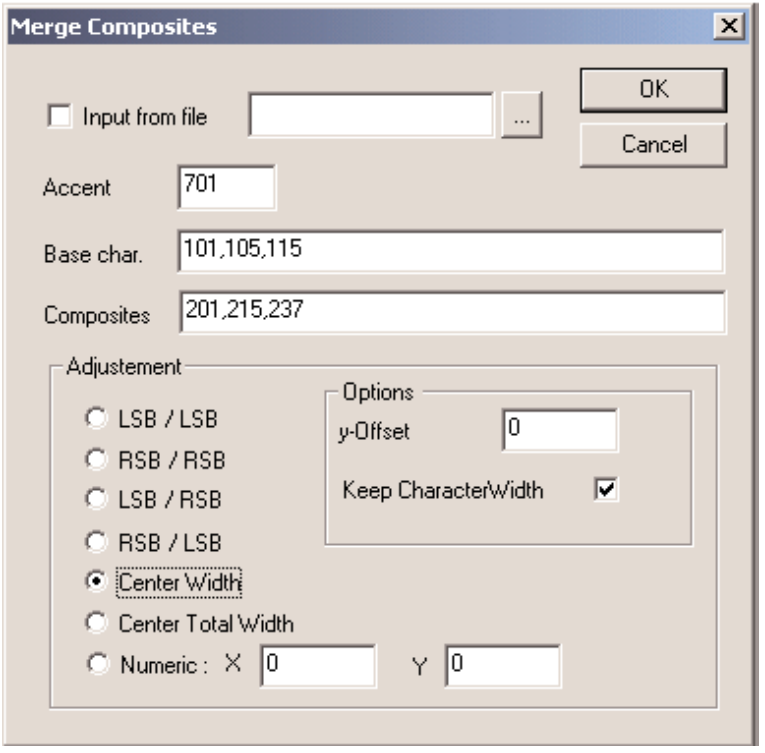
Merge Composites

This is an extremely powerful function that creates accented characters or fractions or other glyphs made out of several composites. It allows a manual parametrization or it can alternatively read and create composites from an external text file.

Selecting the *Merge Composites* function will open a dialog which shows a range of options.

– Accent

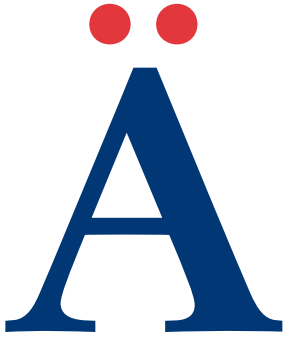
This makes together with the options *Base char.* and *Composites* manual parametrization possible. In the *Accent* input field the Character Number of the accent is specified. For instance number 701 indicates the upper case dieresis.



The Merge Composite dialog.

– Base char.

Here the Character Number(s) must be entered of the character(s) that should be combined with the specified accent. Combining the character numbers 101, 105, 115 which stand for repectively A, E and O with Character Number 701 (Dieresis) will result in Adieresis, Edieresis and Odieresis. Of course these newly generated characters have to be saved at the appropriate positions. Therefore the *Composites* must be specified.



The Adieresis was generated by combining the accent called dieresis with Character Number 701 with base character A, which has character Number 101.

– Composites

Here the Character Number(s) must be entered of the character(s) that are the result of combining the specified accent(s) with the specified base character(s). The standard positions in a IK or BE database for instance for Adieresis, Edieresis and Odieresis are the character numbers 201, 215 and 237. The order of the specified composite Character Numbers must be corresponding with the specified base characters. For instance base character 101 (A) plus dieresis *must* result in Character Number 201 (Adieresis) in case the default Character Layout Files *beeditor.cha* and *urwotf.cha* are used to generate fonts in DTL DataMaster. Details about the Character Layout Files are revealed in Appendix III. More information about the Character Numbers can be found in Appendix IX.

NOTE: The standard places in a IK or an BE database for the lowercase accents are in the range 751–769. The capital accents should be placed in the range 701–719. Normally the lowercase accents are positioned in such a way that no shifting is necessary when they are placed on top of a lowercase character. The same is the case for the capital accents. It is recommended to check first if the capital accents are available and at the right position before generating composite characters.

– *Adjustment*

For the positioning the same options are available as for the *Merge Character* function for individual glyphs from the **Function** menu.

Please note that it is important for merging accents to switch on the *Keep CharacterWidth* option, otherwise the character width of the composite will be different from the base character depending on the adjustement option.

– *Input from file*

Instead of entering all base characters, accents and composites manually the batch mode allows the input from a text file.

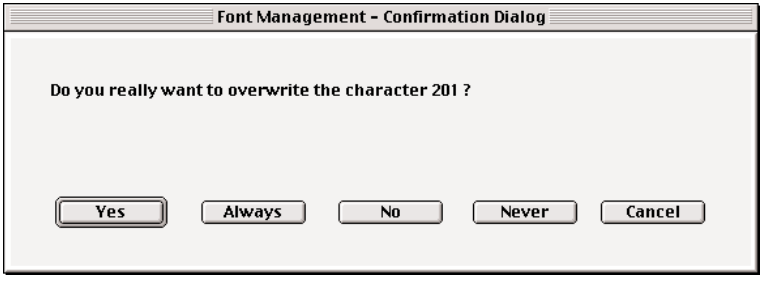
This file should have the following form:

```
// URWNum;URWComp;URWComp
201;101;701
202;101;704
```

The first number is the Character Number of the new composite glyph, the following numbers are of the glyphs which are merged, for instance respectively of the base character and the accent.

It is allowed to specify more than two components, for example to create fractions: 681;623;553;566. In combination with the option *Center Width* and *Keep CharacterWidth* this series of Character Numbers creates a nut fraction (with the Character Number 681) with the width of the first glyph (623 = hyphen) from the three glyphs 623,553 and 566.

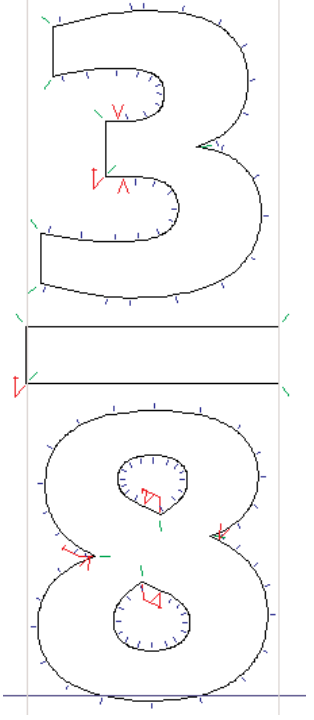
A default text file named *accents.txt* is installed in the same directory as the other DTL FontMaster files. This text file covers all characters containing accents for the Western and Eastern European and Turkish character sets for Mac OS and Windows.



In case the **OK** button has been pushed, the *Font Management-Confirmation Dialog* pops up in case character numbers already exist in the database. *Always* means that all existing characters with the same numbers will be replaced. Please note that overwriting characters is irreversible.

```
// URWNum;URWComp;URWComp
201;101;701
202;101;704
203;101;705
204;101;706
206;101;708
207;101;709
208;101;703
209;101;713
210;103;711
211;103;704
212;103;707
214;104;707
```

The default text file for making composites, accents.txt, is installed in the DTL FontMaster directory.



A nut fraction that was automatically generated using the Merge Composites function from the ***Batch*** *menu.*

Replace Composites

With this function the base characters used for the composites can be replaced by their originals. This is very useful when changes have been made to the base characters after the generation of the composites with the *Merge Composites* function. Please note that the accents are not affected and these have to be replaced manually in case of changes made to the originals.

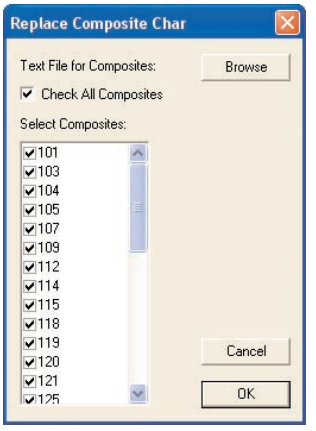
Selecting the *Replace Composites* function will open a simple dialog which shows a small range of options.

– Text File for Composites: Browse

Here a text file that contains composite information, like for instance the default *accents.txt*, has to be selected. The second entries (after the actual Character Numbers of the composites) in the text file will show up in the *Select Composites:* part of the dialog.

– Check All Composites

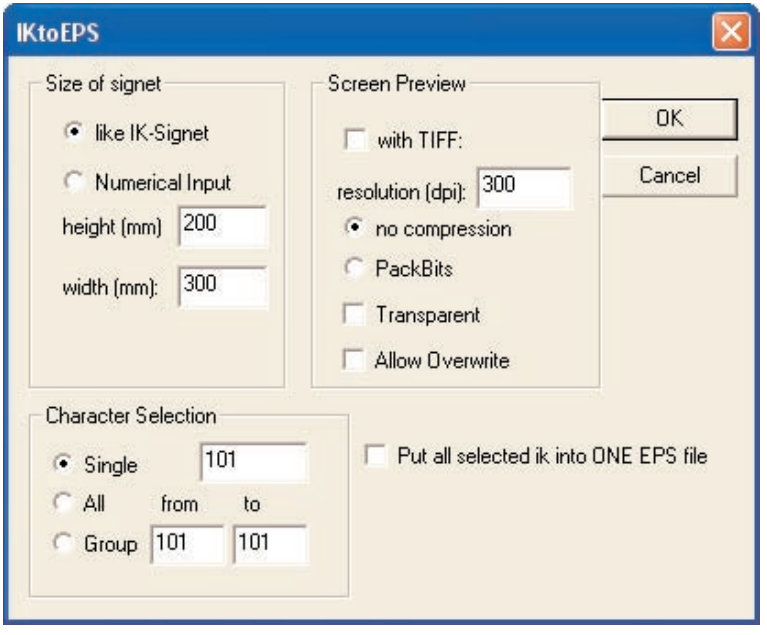
This option makes it possible to replace all base characters listed in the selected text file or, when not activated, to select individual Character Numbers in the *Select Composites:* part of the dialog.



After a text file that contains the composite information has been selected, the Character Numbers of the base characters show up in the dialog.

EPS Output

Glyphs can be exported as EPS data individually or as group. Selecting this batch function will open the *IK to EPS* dialog containing a range of options.



Glyphs can be exported as EPS data using several options.

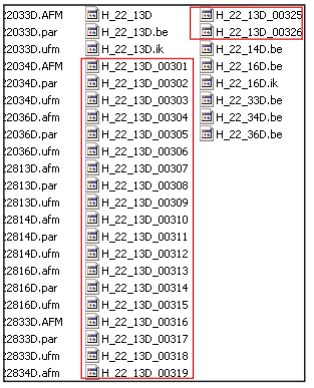
– *Size of signet*

The glyph(s) can be exported with unchanged size by selecting the option *like IK-Glyph*. This preserves the original relation to the other glyphs in the font database, which makes re-importing (for instance after editing in Adobe Illustrator®) possible without any scaling. With the option *Numerical Input* the bounding box can be scaled to any size measured in millimeters.

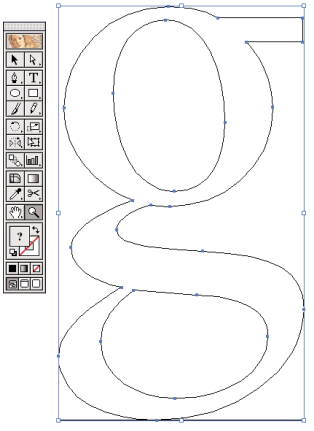
– *Screen Preview*

The EPS files can be provided with a TIFF for screen previewing. As an option the required resolution can be entered. The default reolution is 300 dpi. The TIFF files can be compressed with Apple's *PackBits* format or leaved uncompressed, which is the default setting.

Further options are *Transparant* and *Allow Overwrite*, which let the program overwrite EPS files with the same name. The name of the EPS is taken from the name of the database plus the Character Number. Exporting the glyph with Character Number 302 of the H022013D named database will result in H022013D_00302.EPS. The EPS files are automatically stored in the directory that contains the used font database.



The exported EPS files are stored in the directory that contains the used font database.

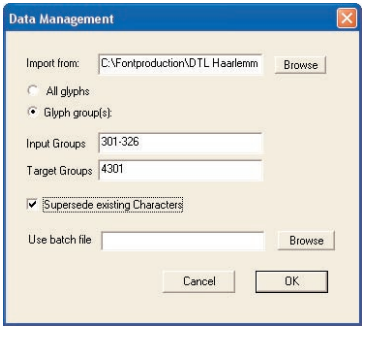


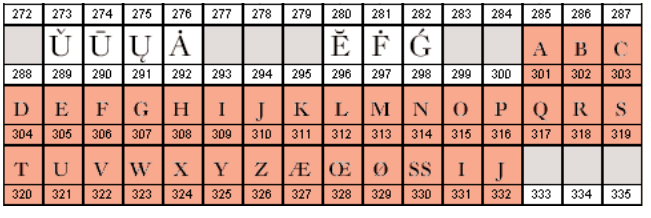
The size of the exported EPS file is based on the bounding box of the glyph.

– *Character Selection*

Glyphs can be exported individually as EPS files but also as group. With the option *Single* a Character Number can be entered. The option *All* will export all the glyphs in the font as EPS files. It is also possible to export ranges using the input fields of the *Group* option.

If multiple glyphs are exported by default the prgram will generate a number of single EPS files. In case one large EPS file that contains all glyphs is required, the function *Put all selected IK into ONE EPS file* should be activated.





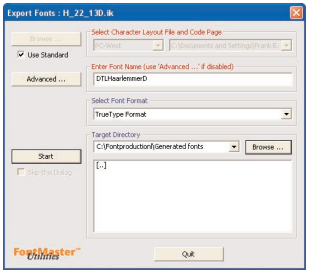
The Import (Data Management) function makes the exchange of data between different files very easy.

Import (Data Management)

With this very powerful batch function glyph data can be imported in the currently open IK database from any other IK database. It is not necesarry

to actually open the other database(s) in DTL BezierMaster. This function makes it for instance possible to build large databases from ones that contain only single code pages, which is very useful especially with the OpenType production in mind.

Using the default Character Layout File *beeditor.cha* when importing for instance PostScript Type 1 fonts in DTL DataMaster will place the lower case characters in the Character Number range 301–326. The same will happen when a Small Caps font is imported because of the used PostScript names for the characters. However, the 'real' names of the small caps are not a,b,c, etc. but Asmall, Bsmall, Csmall, etc. The corresponding Character Numbers in the *beeditor.cha* are 4301, 4302 and 4303. By simply entering the range 301–326 in the *Input Groups* input field and entering 4301 in the input field of *Target Groups*, the small caps will placed in the correct positions in the database, which correspondents with the appropriate PostScript names and Unicode numbers.



The default setting for the Character Layout File in DTL DataMaster will let the program use the beeditor.cha file when PostScript Type 1 and TrueType fonts are converted to the IK format. For OpenType fonts the urwotf.cha file is used by default.

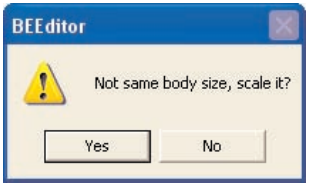
– Import from:

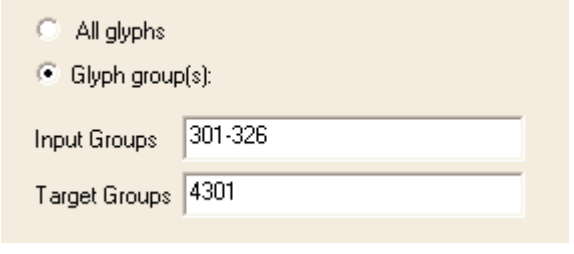
Here the IK database has to be entered from which glyphs have to be imported in the currently active font in the Font Administration tool. You can browse to select the database.

Activating the *All glyphs* option will import all the characters from the selected database. If the *Glyph group(s):* has been activated subsequently the range(s) of the *Input Groups* and the *Target Groups* have to be defined. In the input filed of *Input Groups* the first and last Character Number has to be given divided by a hyphen. For the *Target Groups* only the first Character Number has to be entered. Comma's are used between multiple groups. For example: the groups 101-110, 301-310 entered in the *Input Groups* section can be placed at the *Target Groups* 4101,4301. There is basically no limit to the number of groups defined.

If the *Supersede existing Characters* check box has been activated, existing glyphs with the same Character Numbers as entered in the *Target Groups* input field, will be overwritten. Please pay attention to the fact that these changes are irreverible!

In case the bodysize of the two databases are different, you will be asked whether to scale or not the glyphs to the bodysize of the target database.

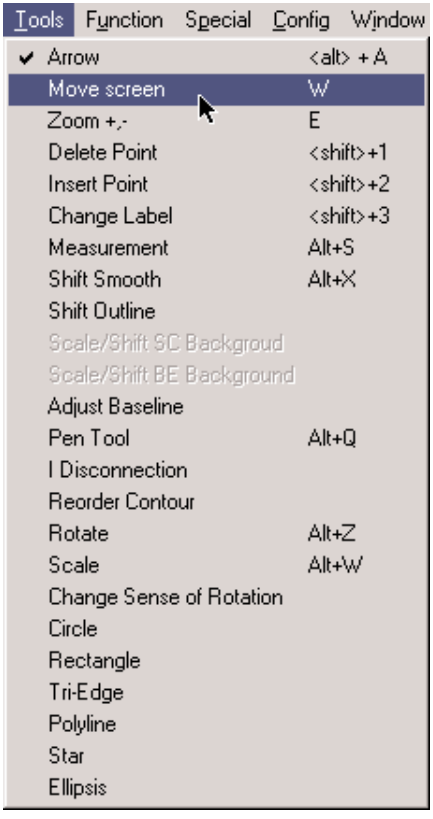




Glyphs can be imported in single or multiple groups.

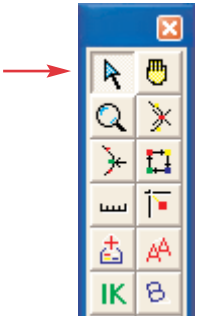
TOOLS MENU

The functions in the **Tools** menu can be chosen either from the pulldown menu or from the *Function Toolbar*. Moving the mouse slowly across the toolbar automatically displays an explanation of the icon. Some functions can also be chosen via keyboard shortcuts. The abbreviations are shown in the pulldown menu on the right side, for example <Alt> + A keys to select the arrow (pointer) tool.



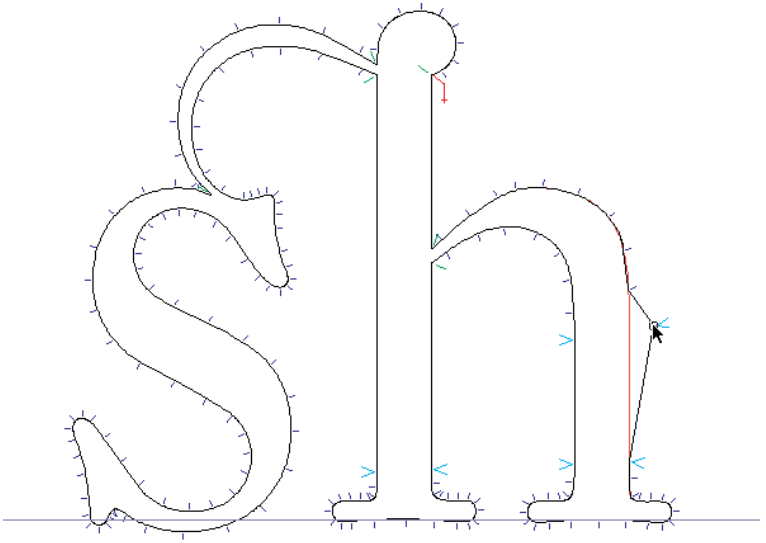
TIP: Several of the functions from the Tools menu, that work in the Character Edit Window only for the active character, can be used for a (large) range of characters using the Batch menu in combination with the Font Administration tool.

Arrow (Space) (⌥Alt + A) (Alt + A)



The arrow or pointer tool is the standard tool to select one or more points or contours as explained in the selection rules:
– Clicking near a point selects this point.
– <⇧Shift>+ mouse click adds a point to the selection.
– A mouse click far away from any point deselects everything.
– A selection with a rectangle can be made by holding down the mouse button and dragging it across the desired area.
– A contour can be selected by double clicking on it.
– A glyph can be selected by double clicking inside it.

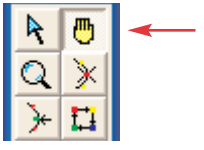
The selected objects can be shifted easily with the arrow tool. Move the arrow near one of the selected objects, press the mouse button, hold it down and move it to the desired position. The objects will be shifted to that position.



*Points can be shifted with the ← and → keys also. The Step Shift units can be defined in the **Config** menu.*

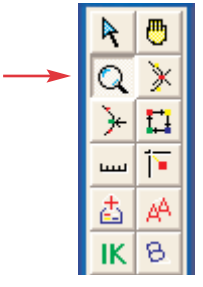
Move Screen (W) (W)

This function allows you to shift the visible display of the currently edited character within the window. It is equivalent to scrolling using the standard scrollbars, but more conveniently.



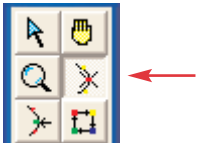
Zoom +,– (E) (E)

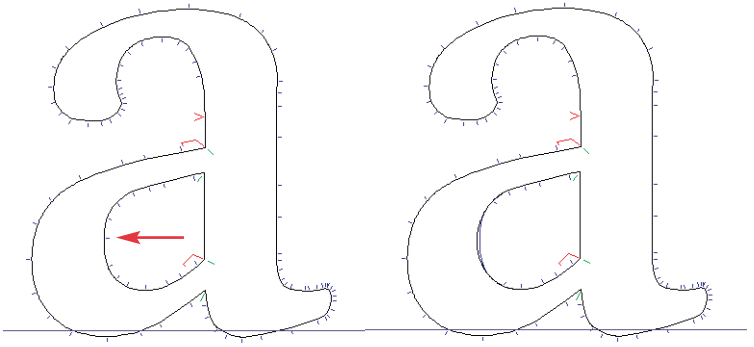
Selecting this function changes the cursor to a looking-glass symbol. You can use it to enlarge or reduce the display.
Zoom (+) Click the mouse button enlarges the display in small steps.
Zoom (-) <⇧Shift>-mouse click reduces the display in small steps.
Holding down the mouse button and dragging the mouse opens a rectangle. If you enlarge the display this rectangle will be the viewable area if the mouse button is released again. If you hold the <⇧Shift> key while dragging the mouse, the currently visible area will be displayed in the rectangle generated by the mouse. **Reset (⌘ + R)** will restore the default image size. The zoomfactor can be set in the **Config** menu.



Delete Point (⇧Shift + I) (⇧Shift + I)

With this function, if you click on an anchor point, it will be deleted immediately. If the anchor point delimites two Bezier curve sections the neighbouring control points will be deleted as well. If you click on a control point both control points of a Bezier section will be deleted.

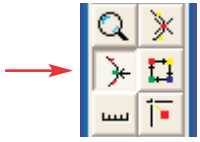




The Delete Point function in action.

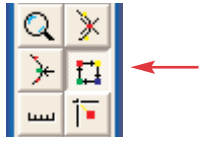
Insert Point (⇧Shift + 2) (⇧Shift + 2)

Click the mouse near the contour to insert a point. By default a curve point will be inserted. <⇧Shift>-mouse click will result in a corner point. <Ctrl>-mouse click produces a tangent point.



Change Label (⇧Shift + 3) (⇧Shift + 3)

This function can also be used to toggle between curve, corner and tangent points. If you use <Ctrl> + mouse click the program will change the label from curve to corner to tangent point and vice versa.

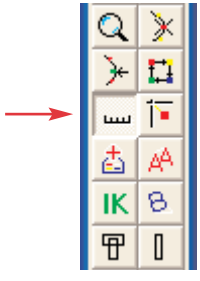


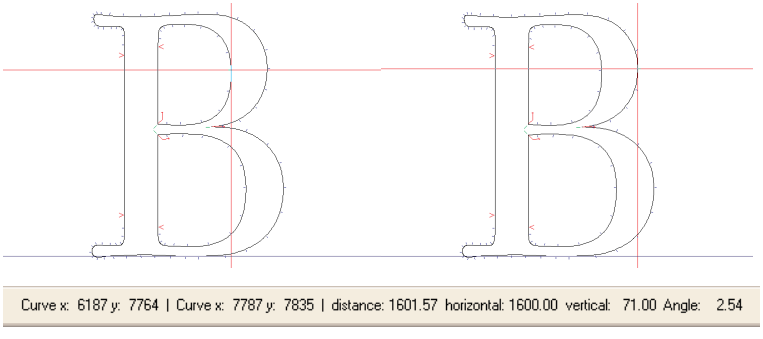
Measurement (Ctrl + s) (Alt + s)

This function can measure the distance between two outline points by simply clicking on them with the mouse. The result will be shown in a small popup window on the screen. You can make the measurement window disappear by selecting another function like *Arrow*.

The measurement window displays the coordinates, the nature (start, curve, corner, tangent). The same information is displayed for the last point which was clicked with the mouse (denoted as 'Another Point').
You will also see the distance of these two points in X and Y, the complete euclidian distance and the angle between these points.

If you click far away from any outline point the screen display position will be displayed and no information about point number etcetera is shown.

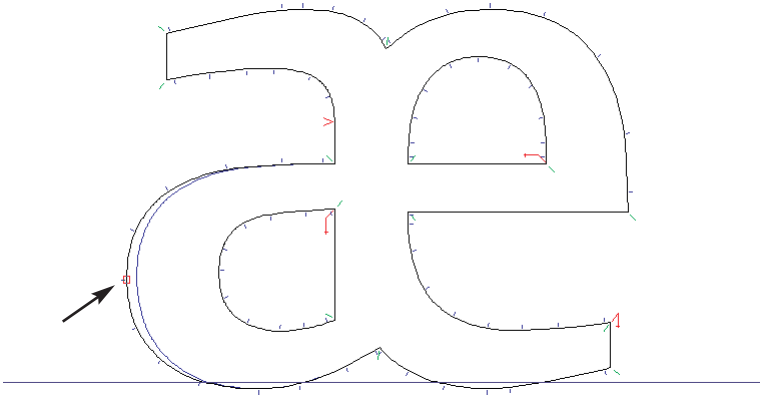


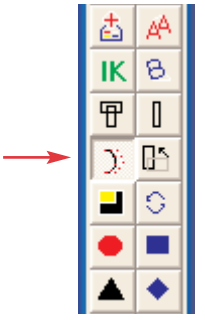


For measuring the distance between two points, select a point with a mouseclick and repeat this with another point.

Shift Smooth (Ctrl + x) (Alt + x)

This function can be used to move points. The point will be shifted such that tangent continuity is preserved. This means that the adjacent points are shifted too



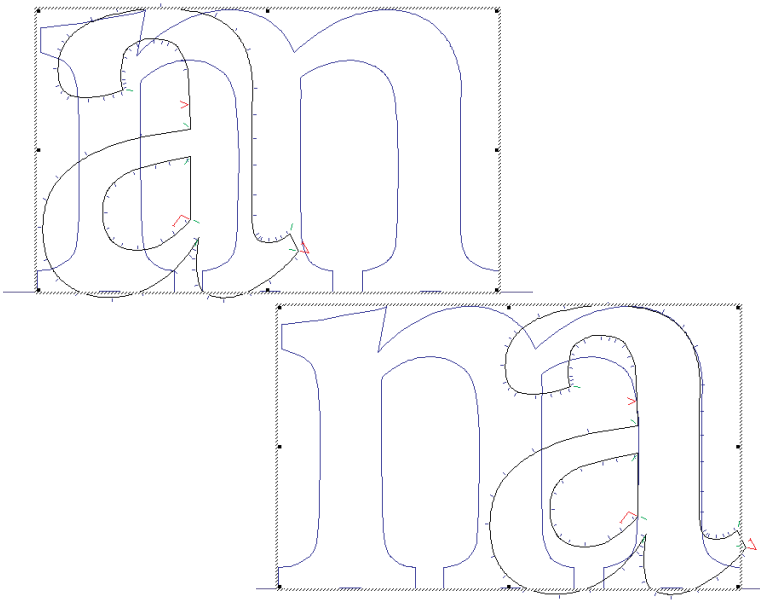


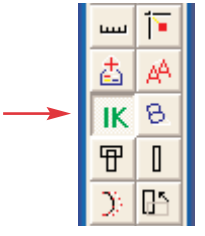
Scale or Shift SC Background

Using this function you can scroll the SC background to a desired position. After selecting this function, a scalable rectangle will appear which you can move on the screen to the desired position.

Scale or Shift IK Background

Using this function you can scroll and scale the IK background to a desired position and size. You have to set the BE background first in the View menu.



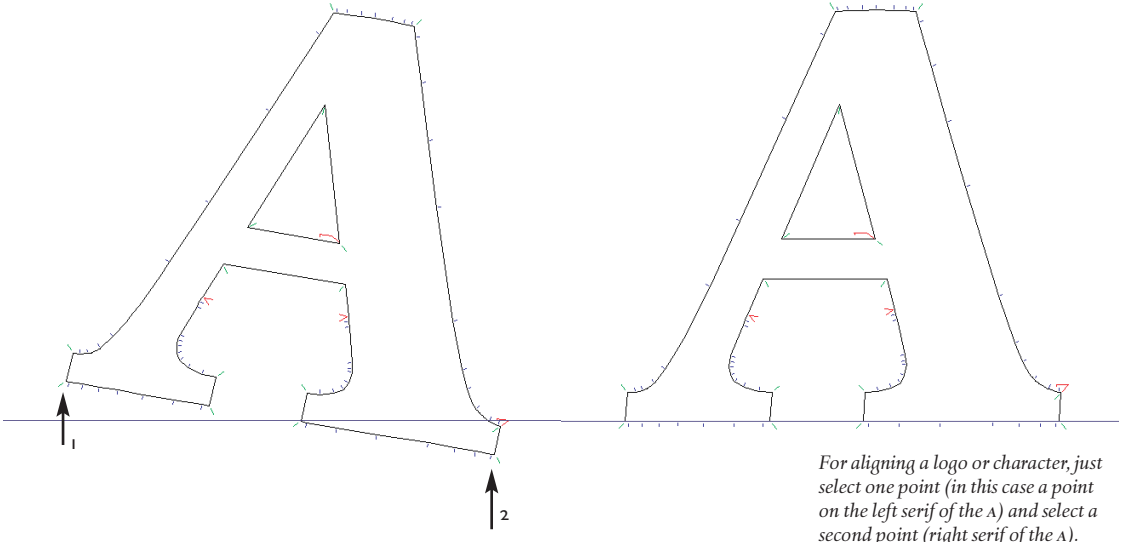


Moving the mouse into the rectangle that marks the IK background will change the cursor into a four-arrow shape. With this cursor activated, the background can be moved. Selecting one of the eight points that mark the background rectangle will change the cursor into a two arrow variation. With this cursor activated, it is possible to resize the background.

Changes made in the background are irreversable.

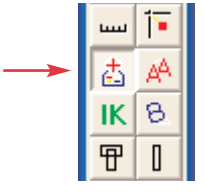
Adjust Baseline

This function is designed to align logos or letters which are automatically converted from scanner data into outline data. It is not possible to align a logo 100 % vertically or horizontally by hand on a scanner. Use this function to align a logo to two outline points. The function will rotate the character so that the two selected points are either horizontal or vertical after the rotation, depending on the angle of the original points. After selecting this function, click on the first anchor or control point and then on the second one.

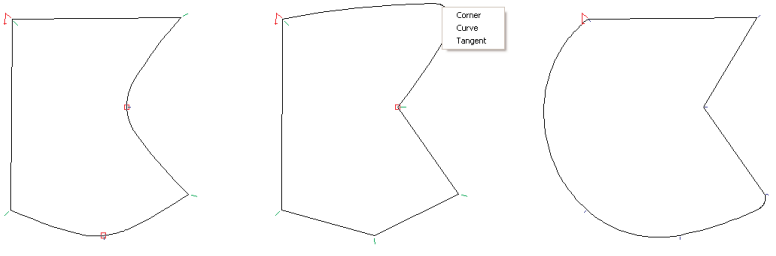


For aligning a logo or character, just select one point (in this case a point on the left serif of the A) and select a second point (right serif of the A).

Pen Tool (Ctrl + Q) (Alt + Q)



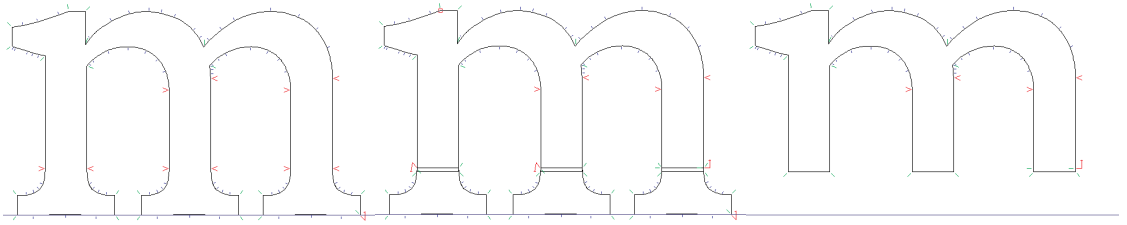
With this function you can draw contours by directly placing points with the mouse in the Character Edit Window. By just clicking and immediately releasing the mouse button, only curve points will be placed. While holding down the <⇧Shift> key and clicking the mouse, the points will be connected by straight lines in vertical or horizontal direction with exception of the line that closes the contour. <Ctrl>-mouse click produces a tangent point. This function makes a perfect combination with the *Change Label* function.

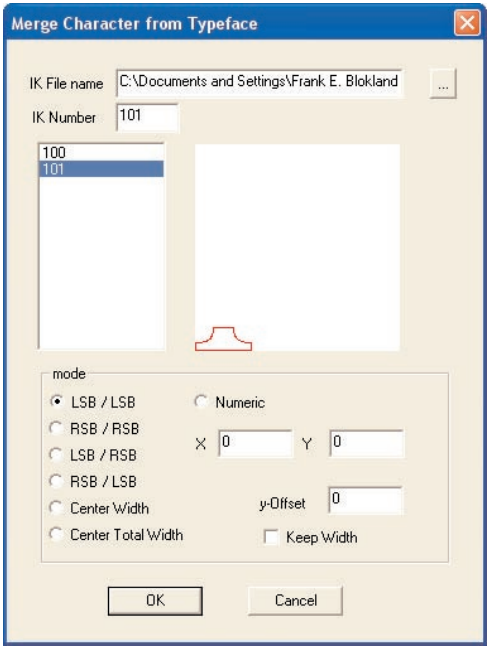


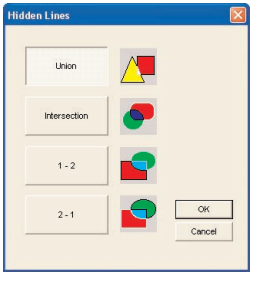
The variations of the shape were made with the Change Label function.

1 - Disconnection

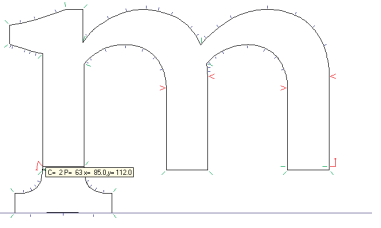
This is a powerfull function to cut a single contour is into two contours with an overlap. Especially in combination with the *Hidden Line* function, the *1-Disconnection* function can be of great value for font production. For instance, a database of different serifs can be built and connected to the stems at a later stage of the production.







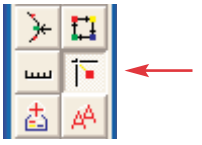
Disconnecting is very easy; just select a point with the mouse and consequently select another point. Two closed contours will be generated automatically. In the illustrations this action is repeated until all three serifs could be removed.



It is possible to build a database of serifs that for instance can be connected numerically to the stems using the Merge Character and Hidden Lines functions.

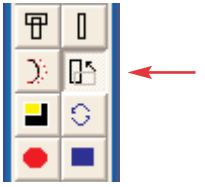
The selection of the location of the cut can be done by selecting two IK points (the function is executed immediately after having clicked onto the second IK point) or by dragging the mouse by holding the left mouse button down over the contour part which has to be separated. The function is executed immediately after having released the mouse button.

The overlap is set in the *Disconector Overlap (units)* option in the **Config** menu.



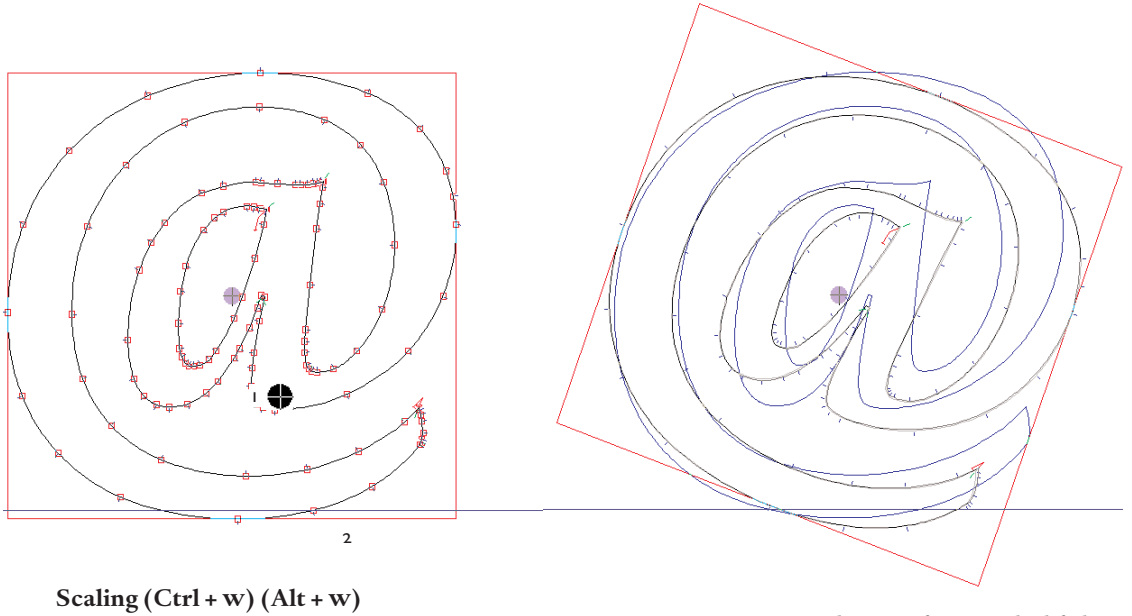
Fit Point to guide line

Using this function, you fit a point ot the nearest horizontal or vertical guideline. Currently not implemented.



Rotate (Ctrl + z) (Alt + z)

Using this function, you can rotate selected contours or the complete character around a defined center of rotation. Set the center of rotation with the mouse. The default setting is the center of the bounding box but you can place the center of rotation practically everywhere. Changing functions in the Function Toolbar wil restore the default position. To rotate the character click on the edges of the rectangle and move the mouse.

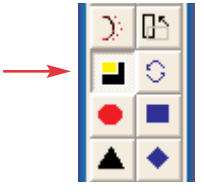


The center of rotation is by default the center of the bounding box (1) but can be placed anywhere else (2). The rotation at the right was made with the second center.

Scaling (Ctrl + w) (Alt + w)

With this function you change the size of characters, contours or contour groups. It is also possible to select individual points. In general scaling can be done either using the mouse or a numerical input via the keyboard.

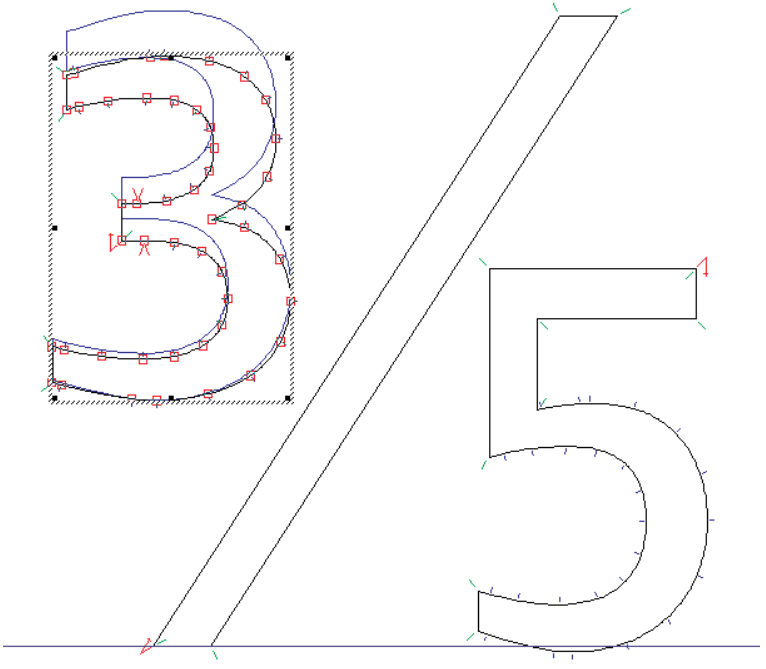
The selection of this function generates a rectangle around the selected points. You can then either scale or shift this rectangle depending on the way you drag the mouse. Selecting certain magic points of this rectangle enables different funtions:



– Diagonally to the upper left side
– Vertically to the upper side
– Diagonally to the upper right side
– Horizontally to the left side
– Central equal to all sides
– Horizontally to the right side

– Vertically to the lower side
– Diagonally to the lower right side

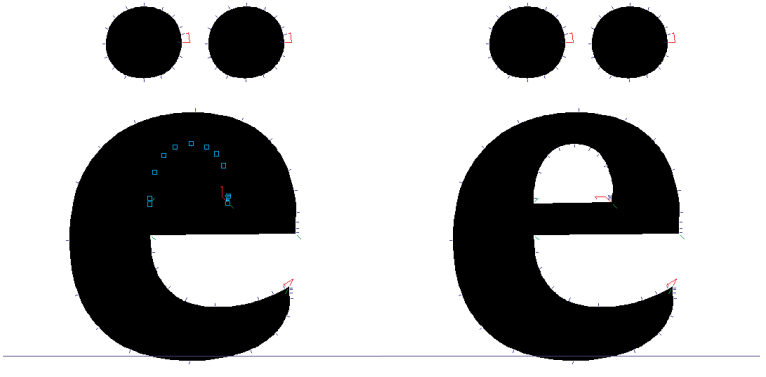
After the direction and factor of the scaling have been selected, execute the function finally by clicking again.

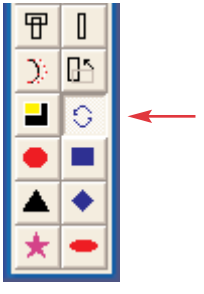


After selecting the points, objects can be scaled in several directions.

Change Sense of Rotation

This function allows you to change the path direction of a contour. Just click on an outline point of a contour to change its sense of rotation. Changing the sense of rotation affects the fill of inner and outer contours.

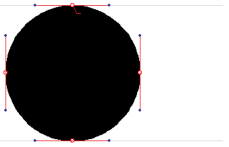




If the direction of the inner contour is not properly set, it will be filled also (left). After changing the sense of rotation this problem will be solved

Circle

This function allows you to create a circle. Select the function, click at a position you wish and drag the mouse. The circle will be generated on the fly and can be changed as long as you hold the mouse button pressed.

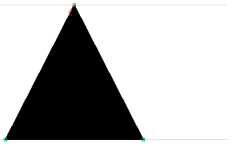


Rectangle

This function allows you to construct a rectangle. It will be generated on the fly and can be changed as long as you hold down the mouse button.

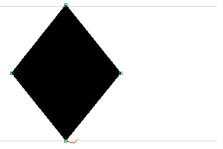
Tri-Edge

This function allows you to construct a triangle. Select the function, click at a position you wish and drag the mouse. The triangle will be generated on the fly and can be changed as long as you hold down the mouse button.



Polyline

This function allows you to construct a polygon. The number of corners for a Polygon can be set in the *Settings* function of the **Config** menu. Select the function, click at a position you wish and drag the mouse. The polygon will be generated on the fly and can be changed as long as you hold down the mouse button.

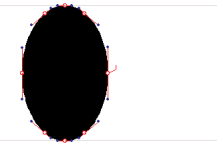


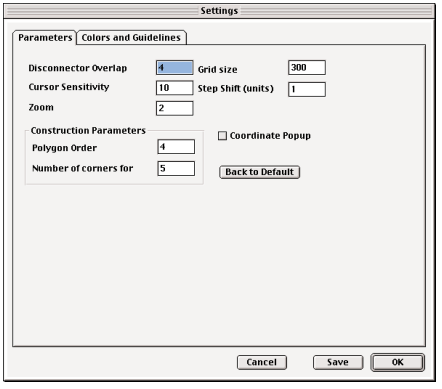
Star

This function allows you to construct a star. The number of corners for a star can be set in the *Settings* function of the **Config** menu. Select the function, click at a position you wish and drag the mouse. The star will be generated on the fly and can be changed as long as you hold down the mouse button.

Ellipsis

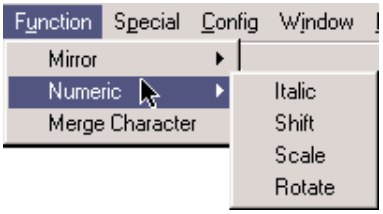
This function allows you to create an ellipsis. Select the function, click at a position you wish and drag the mouse. The ellipsis will be generated on the fly and can be changed as long as you press the mouse button.





The number of corners of the Polygon and Star functions can be set in the Settings function of the ***Config*** *menu.*

FUNCTION MENU

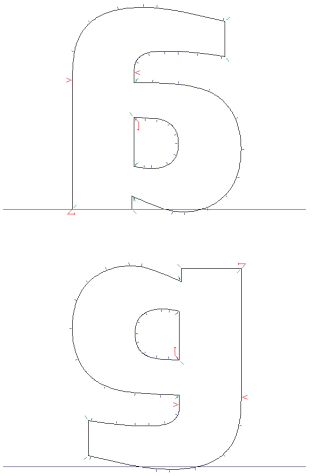


Mirror Left <–> Right

This function mirrors the selected contours or the whole character horizontally around the center of the selected parts.

Mirror Top <–> Bottom

This function mirrors the selected contours or the whole character vertically around the center of the selected parts.



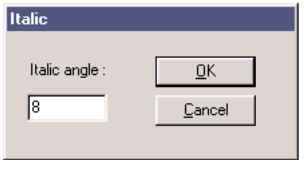
The a on top is mirrored Left <–> Right and the other Top <–> Bottom.

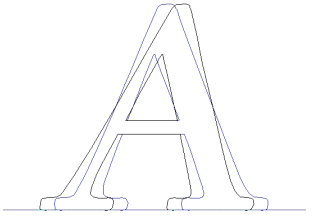
Numeric

Several functions can also be used with numerical input: *Italic*, *Shift*, *Scale*, *Rotate*. Select the contours to be modified, choose the numeric function, for example *Scale*, and then fill in the numeric values (your desired parameters) into the pop-up window, which will appear on the screen.

– *Italic*

Use this function to slant the character electronically. A special selection mode is not required. In this funtion the character mode is always used. After selecting the function a pop-up menu appears. An angle between –45 and +45 degrees is recommended. A positive angle slants to the right.

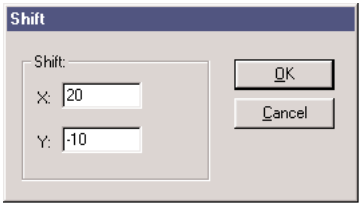


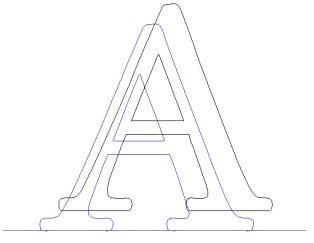


The smallcaps a in this example was slanted to the right.

– *Shift*

This function works on all levels, for selected points, contours or the complete character.

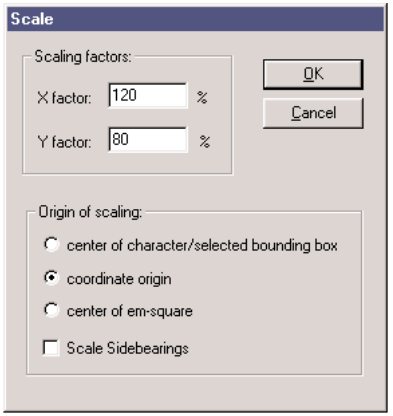


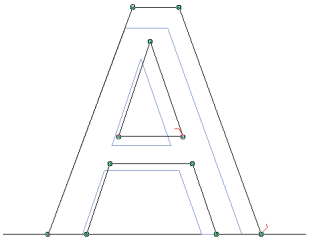


The smallcaps a in this example was shifted over the x- and y-axes.

– *Scale*

After having selected contours or the whole character, input the scaling factors in X and Y direction in %. You can also determine the origin of the scaling and whether or not you will scale the sidebearings simultaneously.

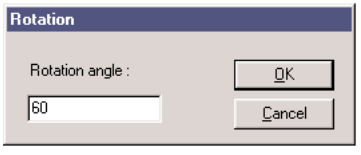


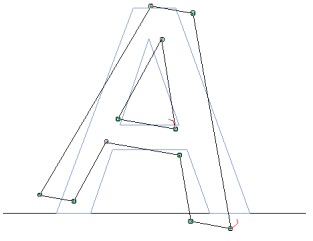


The smallcaps a in this example was scaled 110 % for the X and Y factors with the option coordinate origin selected.

– *Rotate*

The rotate function works for selected contours or the whole character. Input a value in *Rotation angle:* a positive angle rotates clockwise.





The smallcaps a in this example was rotated ten degrees clockwise.

Merge Character

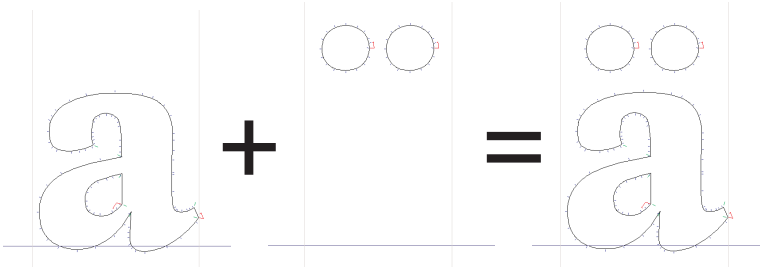
This function allows you to merge another character with the currently edited one. To do so you select this menu option. It allows to select a typeface, from which you can copy a character into your current character. You must select the character number in this typeface and afterwards fix the position for merging.

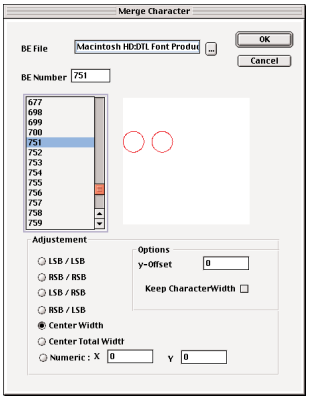
The dialog box allows different options for adjustment, which might be used for different purposes like merging accents or creating fractions or other composite characters. The options are:

– Left side bearing (LSB) / left side bearing (LSB)
– Right side bearing (RSB) / right side bearing (RSB)
– Left side bearing (LSB) / right side bearing (RSB)
– Right side bearing (RSB) / left side bearing (LSB)

– Center Width (center in the Bounding Box)
– Center Total Width (center between the sidebearings)
– Numeric: x, y

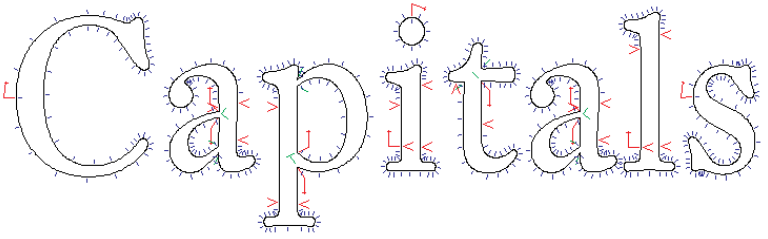
Furthermore there are the options *y-offset* and *Keep CharacterWidth*.





The Merge Character function can for instance be used to place accents.

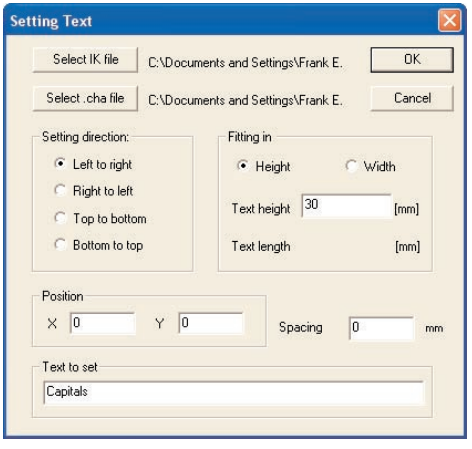
Be aware that to preserve the width of the base character the *Keep CharacterWidth* option must be switched on, otherwise the width of the composite character will differ from the original.



TIP: The merge Character function works only for the active Character Edit Window. To add accents to more than one character the Merge Composites function from Batch menu can be used in combination with a text file that describes which glyphs have to be combined.

Set Text

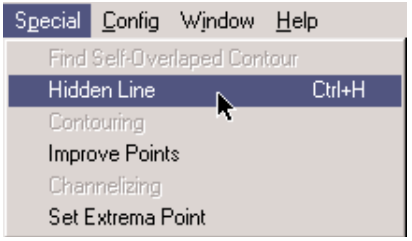
With this function text can be set in the Character Edit Window. Any IK database can be chosen and a range of options is available to control the output. This function is especially useful for the creation of logos.



There are several options in the Setting Text dialog for controlling the output in the Character Edit Window.

SPECIAL MENU

The pulldown menu shows the following options.

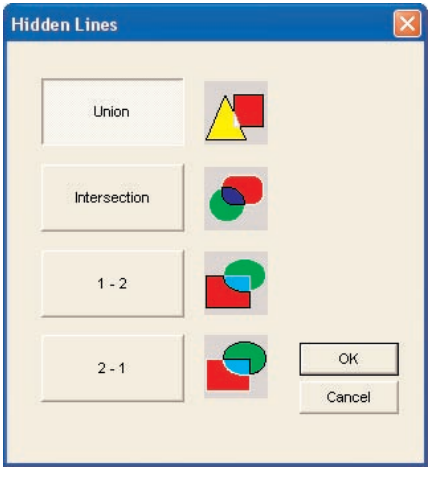


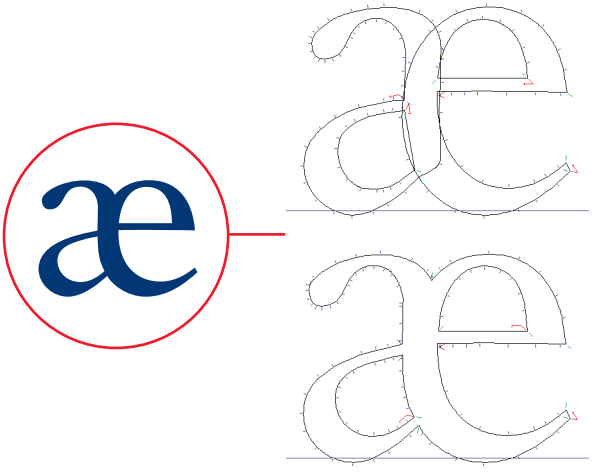
Find Self-Overlapped Contour

The location of this function has been moved into 'Improve points' in this menu.

Hidden Line (⌘ + H) (Ctrl + H)

This function, which is also could have been named *Remove overlap*, merges overlapping contours. It currently works always on the complete character.





The Union option merges the overlapping contours

– *Union*

Merges the contours.

– *Intersection*

Creates the intersection. Only the overlapping parts remain. The rest of the contours are deleted.

– *1-2*

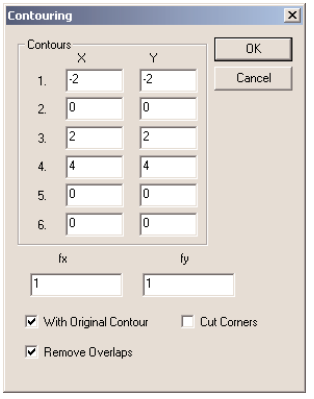
Deletes the second contour and the part of the first contour that was overlapped.

– 2-1

Deletes the first contour and the part of the second contour that was overlapped.

Contouring

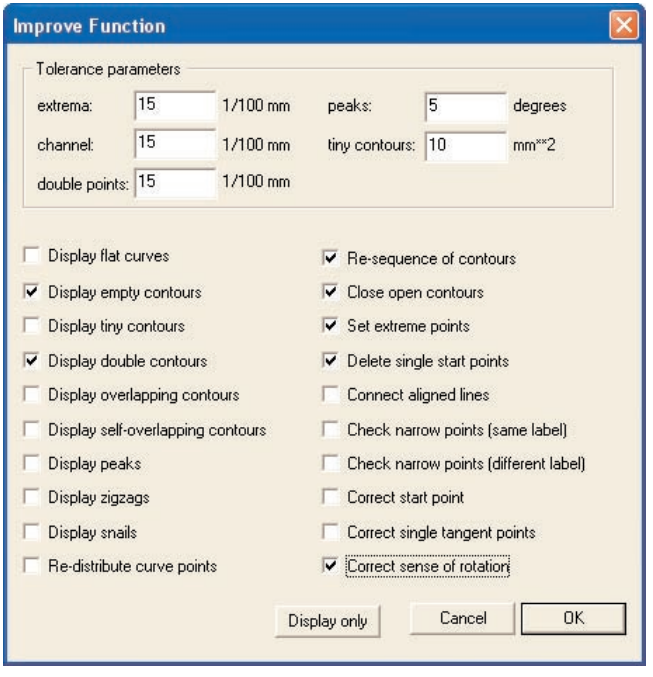
This function is used to create additional contours automatically, so-called outlined characters. It can also be used to bolden a typeface or make it thinner. Input up to 6 values in mm into the menu to create up to 6 additional contours. A positive value of for example 2 adds a contour with 2 mm distance from the original contour to the outside. A negative value works to the inside.

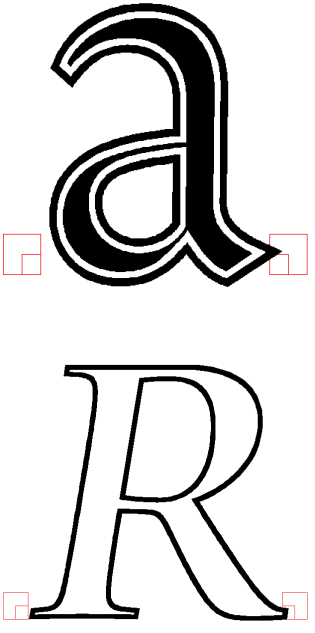


*Up two six contours can be generated using the Contouring function from the **Special** menu.*

Improve Points (⌘ + I) (Ctrl + I)

This function improves the Ikarus outline if necessary. Certain problematic digitization features can be removed and corrected, as listed in the popup menu below. You can specify certain parameters which govern the determination of the digitization errors.





Two examples of contouring. The outline of the capital R was made with different values for the horizontal (slightly thinner) and the vertical lines.

– *Display flat curves*

Shows curve points which can only be regarded as straight from the Ikarus algorithm. These curve points are marked with a blue circle.

– Display empty contours

Shows contours, which consist out of less than four digitizations. The start point of such a contour is marked with an olive square.

– Display tiny contours

Shows contours, which areas are smaller than the *tiny* tolerance. The start point of such a contour is marked with a fair brown square.

– Display double contours

Shows pairs of contours, which digitizations are nearly identical. This kind of mistake is sometimes produced during manual digitization. All digitizations of such contour pairs are marked with dark red circles.

– Display overlapping contours

Shows the overlap of overlapping contours. The overlaps of such contours are marked with a violet square.

– Display self-overlapping contours

Shows overlap within a single contour. The overlaps of such contours are marked with a red square.

– Display peaks

Shows peaks (edge points) whose angle is smaller than the *peaks* tolerance. The peaks are marked with a fair brown square. A recommended value is ten degrees.

– Display zigzags

Shows parts of the contour which have the shape of a zigzag smaller than the *double points* tolerance. This kind of mistake is sometimes produced during manual digitization. The zigzags are marked with a yellow square.

– Display snails

Shows contours where the digitization goes further than the start/end point. This kind of mistake is sometimes produced during manual digitization Such contours are marked with a fair red square at the position of the start point.

– Re-distribute curve points

Tries to re-distribute the curve points in curves, so that the tangents between them are 30 degrees. There is no marking.

– Re-sequence of contours

Sorts the contours regarding the IK rules, which means outer contours get a lower contour number than the inner ones. There is no marking. **Re-**

TIP: The Improve Points function works only for the active Character Edit Window. To check and improve all characters of a font database DTL ContourMaster can be used. The options in this programme are comparable to these in the Improve Points function.

– Close open contours

This function closes open contours. After correction of such a contour a green circle will mark the start point.

– Set extreme points

Moves curve points, which are located near an extreme point onto the exact position of the extremum. If there is no curve point within the *extrema* tolerance, a new curve point will be inserted onto the extremum. There is no marking.

– Delete single start points

Deletes start points which are the only digitization of its 'contour'. A blue circle will mark the (former) position of the deleted start point.

– Connect aligned lines

This function checks two connected straight lines and unifies them to one straight line, if they only differ within the *channel* tolerance and within an angle of ten degrees. A brown circle marks the position of the deleted point.

– Check narrow points (same label)

Deletes one of two neighbored IK points (the one with the higher digitization number) of the same label in case the two points are located closer to each other than the *double points* tolerance. An olive circle will mark the (former) position of the deleted point.

– Check narrow points (different label)

Deletes one of two neighbored IK points (the one with the higher digitization number) of any differing label in case the two points are located closer to each other than the *double points* tolerance. An olive circle will mark the (former) position of the deleted point.

– Correct start point

Corrects wrongly located start points. This kind of mistake is sometimes produced during manual digitization. There is no marking.

– Correct single tangent points

Converts tangent points without edge or tangent as neighbor into a curve or edge point, depending on the neighbors. A fair-violet circle will mark the position of the point with the changed label.

– Correct sense of rotation

Corrects the sense of rotation of contours, which means outer contours will turn clockwise and inner ones will turn counter-clockwise. A fair-red circle will mark the position of the start point of the corrected contour.

CONFIG MENU

Function Settings: Parameters

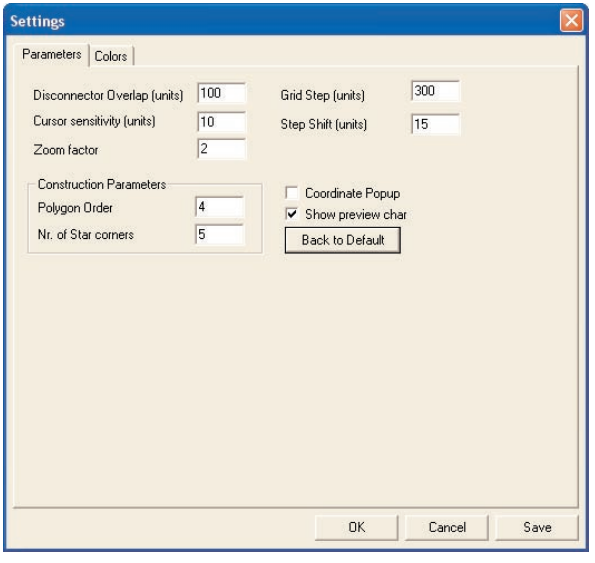
In the **Config** menu *Function Settings: Parameters* you can set different parameters:

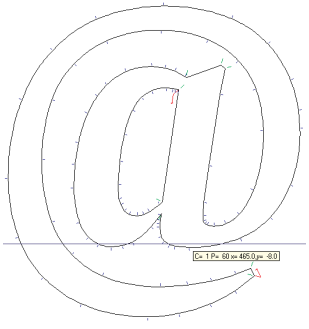
Function	*Default value*
Disconnector Overlap	100 (units, 1/100 mm)
Cursor Sensitivity	10 (units, 1/100 mm)
Zoom	2 (factor)
Grid size	300 (units, 1/100 mm)
Step Shift	15 (units, 1/100 mm)

Please adjust these parameters if the bodysize of your font is not 15000 (units), which is the default setting. The basic idea behind the 15000 units is, that a high resolution of the database means a greater control over the details in the design and also a relatively small loss of quality when the resolution is scalled down for the generation of PostScript Type 1 and OpenType (CFF) fonts (1000 units) or TrueType fonts and OpenType (TTF) fonts (2048 units).

The construction parameters determine the number of corners of a polygon and a star.

If the *Coordinate popup* option is enabled, then the coordinates of the point nearest to the cursor will be shown in the Character Edit Window.





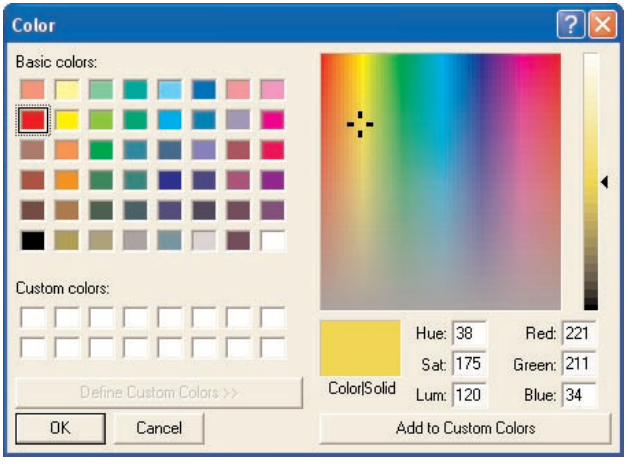
If the Coordinate popup option is enabled, the coordinates of the point nearest to the cursor will be shown.

Function Settings: Color

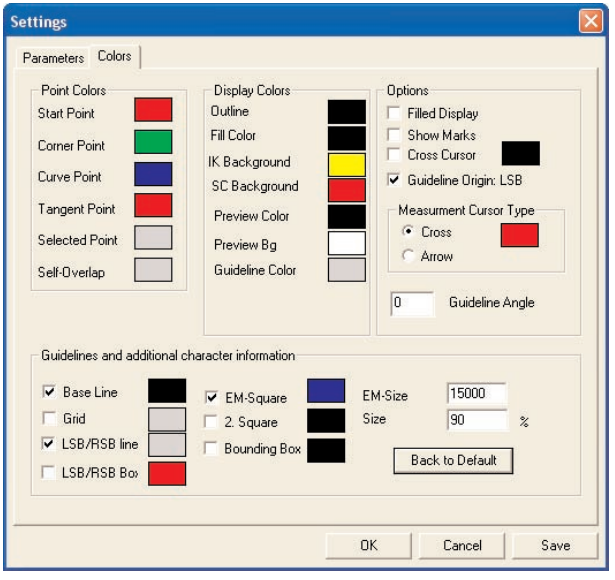
In the **Config** menu *Settings: Color* you can set the colors and the guidelines which shall be shown or not.
The meaning is mostly self-explaining:
X minimum / X maximum
Y minimum / Y maximum
Base Line (at Y position: 0)
Grids (Grid distance can be set in the Parameters menu above)
LSB / RSB line = Left side bearing / Right side bearing
EM-size (standard IK-bodysize = 15000 (1/100 mm))
Cross Cursor (shows a crosshair cursor instead of an arrow)

Marks: Colors (default)
Start point *red*
Corner point = Anchor point *green*
Curve point = Control point *blue*
Tangent point = Smooth Anchor point *red*
Outline color *black*
SC background *red*
Selected point *grey*
Fit to guide *grey*
Self-overlap *grey*
IK background *yellow*
Preview Text Color *black*
Preview Background Color *white*

Just double click on the colors to change them.



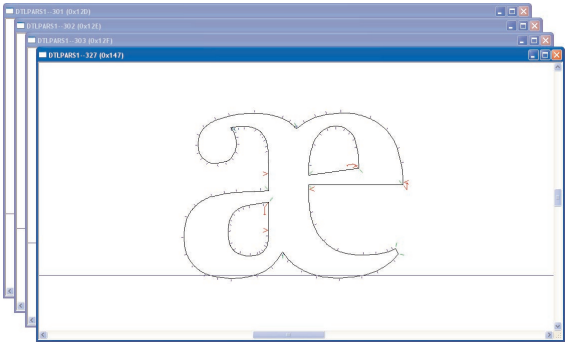
Arbritary colors can be selected.



WINDOW MENU

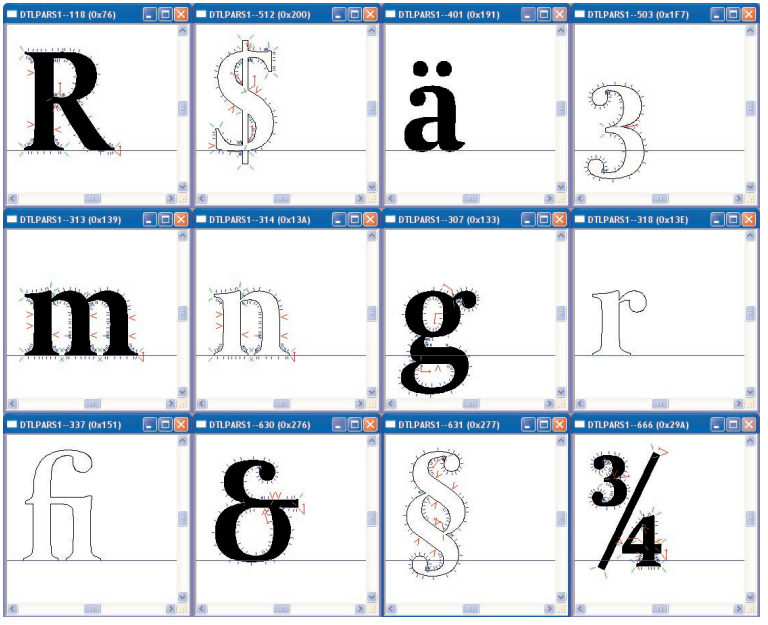
Cascade

When more than one Character Edit Window is opened, they overlap each other. By choosing *Cascade* they are ordered in cascading windows that can be moved to the foreground by clicking them.



Tile (⌘ + J) (Ctrl + J)

By choosing *Tile* all opened Character Edit Windows are placed in a tile pattern.



All open characters are placed in a tile pattern.

Close all windows

When this function is chosen, all Character Edit Windows are closed. The Character List Window remains opened.

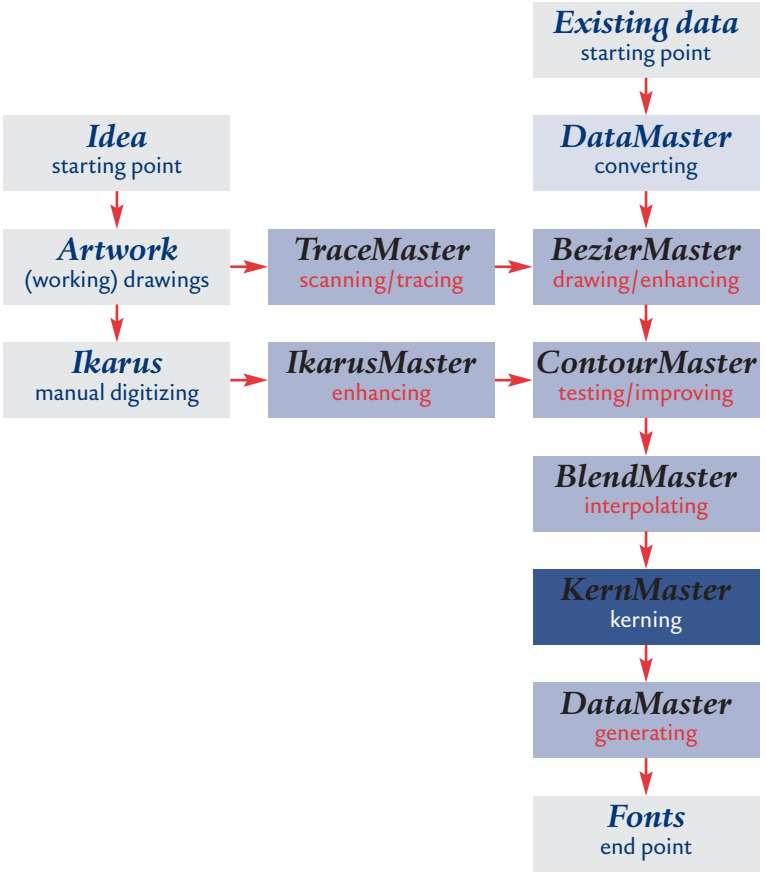
Function	Mac OS	Windows
Arrow	(Space) (⌥Alt + A)	(Space) (Alt + A)
Background on/off	(Ctrl + B)	(Alt + B)
Background chars on/off	(⌘ + B)	(Ctrl + B)
Change Character Header	(⌘ + I)	(Ctrl + I)
Change Font Header	(⌘ + ⇧Shift + F)	(Ctrl + ⇧Shift + F)
Change Label	(⇧Shift + 3)	(⇧Shift + 3)
Close	(⌘ + W)	(Ctrl + W)
Copy	(⌘ + C)	(Ctrl + C)
Copy into Background	(⌘ + Ctrl + C)	(Ctrl + Alt + C)
Cut	(⌘ + X)	(Ctrl + X)
Delete character	(← Backspace)	(Delete)
Delete Point	(⇧Shift + I)	(⇧Shift + I)
Digitizing Number	(⌥Alt + D)	(Alt + D)
Display Marks	(⌘ + M)	(Ctrl + M)
Edit Coordinates	(⌘ + E)	(Ctrl + E)
EM Square on/off	(⌘ + D)	(Ctrl + D)
Exit	(⌘ + Q)	(Ctrl + Q)
Fill Color	(⌘ + F)	(Ctrl + F)
Font Administration	(⌘ + U)	(Ctrl + U)
Grids	(⌘ + G)	(Ctrl + G)
Hidden Line	(⌘ + H)	(Ctrl + H)
Horizontal Guidelines	(⇧Shift + H)	(⇧Shift + H)
Improve points	(⌘ + I)	(Ctrl + I)
Insert Point	(⇧Shift + 2)	(⇧Shift + 2)
Measurement	(Ctrl + S)	(Alt + S)
Metrics Editor	(⌘ + K)	(Ctrl + K)
Move Screen	(W)	(W)
New	(⌘ + N)	(Ctrl + N)
Next Character	(⌘ + →KeyRight)	(Ctrl + →KeyRight)
Open	(⌘ + O)	(Ctrl + O)
Paste (with offset)	(⌘ + V)	(Ctrl + V)
Paste (without offset)	(⌘ + ⇧Shift + V)	(Ctrl + ⇧Shift + V)
Paste from Background	(⌘ + Ctrl + V)	(Ctrl + Alt + V)
Pen Tool	(Ctrl + Q)	(Alt + Q)
Previous Character	(⌘ + ←KeyLeft)	(Ctrl + ←KeyLeft)
Print	(⌘ + P)	(Ctrl + P)
Print Options	(⌘ + ⌥Alt + P)	(Ctrl + Alt + P)
Print Setup …	(⌘ + ⇧Shift + P)	(Ctrl + ⇧Shift + P)
Redo	(⌘ + Y)	(Ctrl + Y)

Function	Mac OS	Windows
Replace by Background	(⌘ + Ctrl + R)	(Ctrl + Alt + R)
Reset	(⌘ + R)	(Ctrl + R)
Rotate	(Ctrl + Z)	(Alt + Z)
Save	(⌘ + S)	(Ctrl + S)
Save as	(⌘ + ⇧Shift + S)	(Ctrl + ⇧Shift + S)
Scaling	(Ctrl + W)	(Alt + W)
Second EM Square on/off	(⌘ + ⇧Shift + D)	(Ctrl + ⇧Shift + D)
Select all points	(⌘ + A)	(Ctrl + A)
Select Background	(⌘ + ⇧Shift + B)	(Ctrl + ⇧Shift + B)
Shift smooth	(Ctrl + X)	(Alt + X)
Tile	(⌘ + J)	(Ctrl + J)
Undo	(⌘ + Z)	(Ctrl + Z)
Vertical Guidelines	(⇧Shift + V)	(⇧Shift + V)
V/H Guide Lines	(⌘ + ⇧Shift + L)	(Ctrl + ⇧Shift + L)
Zoom +,–	(E)	(E)

Dutch Type Library

DTL KernMaster

's-Hertogenbosch / Hamburg
Summer 2004

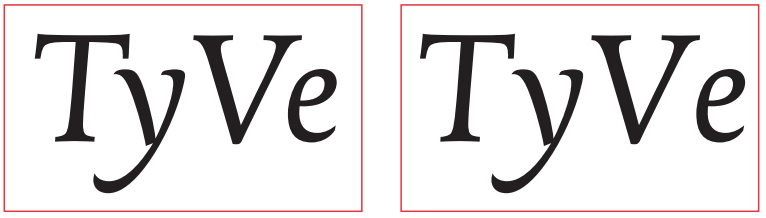


The diagram shows a typical workflow based on the modules of DTL FontMaster.

DTL KernMaster is the module for the automatic creation of kerning pairs for BE and IK databases with output into AFM file or kern.fea file. The program is especially developed with OpenType production in mind: it is possible to generate kerning for a complete BE or IK database, including Cyrillic, Greek, Eastern-European, etc. at once. Class kerning is supported with output into kern.fea files for the OpenType production. The module also supports batch processing.

Kerning should not be used to correct but to refine the spacing and the fact is, that there are numerous combinations which you can't get optimal without kerning, especially when Eastern European, Cyrillic and Greek character sets are included. Using very sophisticated algorithms, DTL KernMaster makes it possible to perfectly optimize the spacing.

The quality of the kerning generated by DTL KernMaster is very high but changes can always be made manually in the Metrics Editors of DTL Bezier- and IkarusMaster. Also the class kerning can be edited here.

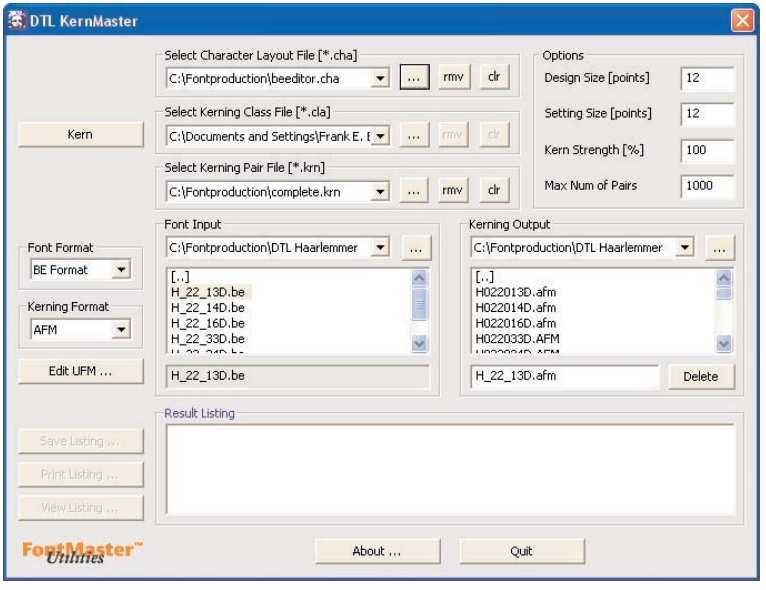


Two versions of the same typeface, respectively with and without kerning pairs calculated by DTL KernMaster.

Starting DTL KernMaster

In DTL KernMaster the glyphs are converted first to extremely high resolution bit maps before sophisticated algorithms are used to calculate the kerning. Complete BE and IK font databases can be supplied with kerning pairs. There is basically no limit to the number of kerning pairs, so very large databases that support multiple scripts can be used in combination with extensive kerning pair definitions. The actual calculation is a matter of seconds, depending on the speed of the CPU, of course.

DTL KernMaster does not work with font formats like PS Type 1, TrueType or OpenType directly. Fonts have to be converted to the BE or IK format in DTL DataMaster first, before using DTL KernMaster.



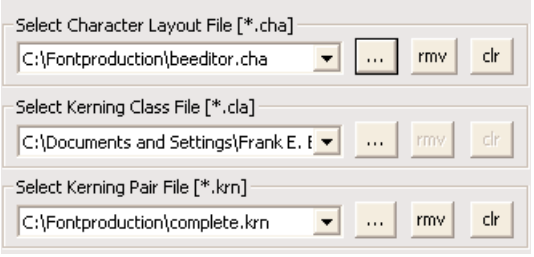
All basic functions can be handled in the main dialog of DTL KernMaster.

Main dialog

The main dialog, which is shown when the program starts, is devided into four sections:

– One that contains info about the supporting Character Layout file, Kerning class file and Kern Pair file.

– A section that contains the options.
– A section that contains the font input and the kerning output. Also the font format and the kerning format can be selected here.



– And to complete the dialog, there is also a listing of the kerning process.

```
# -- Kerning classes
@A_LC = [a aacute acircumflex adieresis agrave
aring atilde abreve amacron
aogonek];
@A_SC = [Asmall Agravesmall Aacutesmall
Acircumflexsmall Atildesmall
Adieresissmall
        Aringsmall Abreve.small Amacron.small
Aogonek.small];
```

Kerning class info as defined in the default kernmaster.cla file.

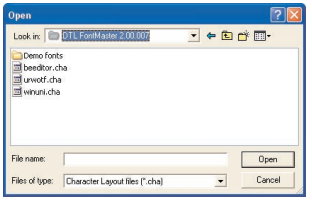
1. Supporting (text) files

Before kerning can be calculated for a BE or IK font database, supporting text files have to be selected.

1.1 *Select Character Layout File [*.cha]*

The selected Character Layout File should correspond with the glyph arrangement of the database. By default the beeditor.cha file is selected but any other Character Layout File can be used in case for instance the beeditor.cha file does not support all the scripts in the font database.

If the numbering system used in the database differs from the listing in the Character Layout File, the resulting kerning info will be unpredictable and unreliable. More information about the syntax of the Character Layout File can be found in Appendix III.



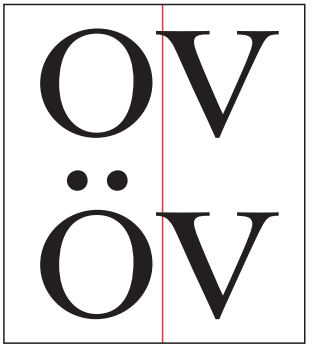
NOTE: *The arrangement of the glyphs in the database must correspond with the selected Character Layout File, otherwise the resulting kerning info will be unreliable.*

1.2 *Select Kerning Class File [*.cla]*

For the production of kern feature files kerning classes can be used. A kern class is a group of characters that all should be handled the same way by an application if kerning has been applied for a combination of the kern class with another character (or kern class). This way the structure of a file that contains the kern pairs as kern classes differs from an AFM file, in which all pairs are defined individually. However, files that support kern class pairs also can contain individual combinations. The kerning pairs listing of separate characters and/or kern classes, is defined in **.krn* files.

The kern classes are defined in a Kerning Class File (*.cla), by default the *kernmaster.cla* file. In this file the groups that form the classes are listed. As with all other supporting files for DTL FontMaster, also a Kerning Class File can be altered using a simple text editor.

Although the Kerning Class File currently has no influence at all on the AFM file output, a Kerning Class File should be selected always otherwise an error message will be shown by the program.



Kern class info assures the same handling of kerning for all members of the same kern class.

1.3 *Select Kerning Pair File [*.krn]*

The Kerning Pair File contains a listing of all the kerning pairs in a simple format. This is completely comparable with the definitions in an AFM file: *KPX [PostScript name] [PostScript name]*. For the kern feature file output, in stead of the PostScript name the name of the kern class should be provided. The name of the kern class should be exactly according to the definition in the Kerning Class File (*.cla). For example: if the kern class has been defined as follows: *@A_LC = [a aacute acircumflex adieresis agrave aring atilde abreve amacron aogonek]*, in the kern pair this class should be represented by *@A_LC*.

```
Comment DTL KernMaster version 2.00.009
Comment Copyright Dutch Type Library, 2004
Comment Creation Date:17/03/04
StartFontMetrics
StartKernData
StartKernPairs 4
KPX @A_UC @C_UC
KPX T e
KPX one one
KPX @A_UC v
EndKernPairs
EndKernData
```

An example of a small Kerning Pair File that contains kern class information.

The default Kerning Pair File is *kernmaster.krn*, which contains roughly 1000 kerning pairs for all Western European languages. Besides this kerning file also a file called *complete.krn* has been placed by the installer in the same directory as the DTL FontMaster program files. This file contains roughly 5000 kerning pairs for Western European, Eastern European, Turkish, Cyrillic and Greek character sets. Using the *complete.krn* file in combination with a font database and Character Layout File that both cover multiple scripts, will result in an extensive AFM file which can be used for the generation of fonts that either support single or multiple code pages.

Шрифты

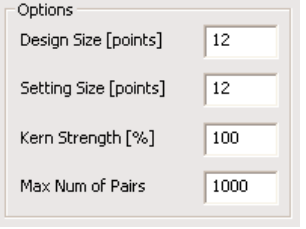
If the font database contains multiple scripts, all kerning pairs can be calculated at once.

2. Options

There is a range of options to influence the calculated kerning values.

2.1 *Design Size [points]*

The point size entered here interacts with the Setting Size and this results in a certain tightness of the kerning. The point size is defined in the digital pica points (pt) where 1 pt measures 0.351 mm. The Design Size indicates the point size for which the typeface was originally designed and spaced. The default of 12 points indicates that the typeface was designed and spaced for text purposes. The same point size is the default setting for the Setting Size. Normally there is no need to alter these settings. The effect on the calculated kerning values is always based on the difference between the given point sizes for the Design Size and the Setting Size.

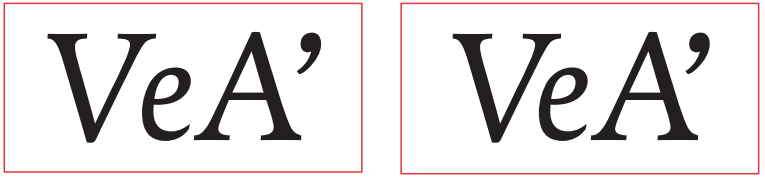


There are several options to control the tightness i.e. 'kernstrength' of the kerning.

2.2 *Setting Size [points]*

If the Setting Size is smaller than the Design Size, the negative kerning values will be smaller and the positive kerning values larger, so the overall image will be looser. If the Setting Size is larger than the Design Size there will be more negative kerning and less positive kerning, so the overall image will be tighter. The given point sizes only indicate the relative difference in kerning; there is no need at all to enter the exact point size for the Design Size if this is for instance unknown.

NOTE: The entered point sizes are only used to control the kerning values. There is no need to enter an exact value for the Design Size.



The tighter kerning in the example at the right was achieved by entering a larger point size for the Setting Size.

Normally a text face will be somewhat wider spaced in comparison with a display typeface. Choosing tighter kerning through entering a larger number for the Setting Size will affect the overall spacing. Please note that the actual spacing of the font is not changed, of course.

2.3 *Kern Strength [%]*

With Kern Strength the tightness of the kerning is fixed. Basically the effect is comparable with what can be accomplished with Design Size and Setting Size. However the difference is that Kern Strength works with percentages and has to be controlled with an integer between 10 and 200. A value smaller than the default 100 will result in a wider kerning and a value larger than 100 will result in a tighter kerning.

With an integer between 10 and 200 the tightness of the kerning can be controlled. The default value is 100.

The extra functionality that this option offers, is that it can be combined with Design Size/Setting Size. This way for instance a standard set narrow or wide kerning can be used in combination with optimized solutions for the different point sizes.

2.4 *Max. Num of Pairs*

There is no technical limit to the number of kerning pairs. The default amount is 1000 but any number can be entered. The standard *kernmaster.krn* file contains roughly 1000 kerning pairs and is limited to Western European character combinations. The *complete.krn* file contains also combinations for Eastern European, Turkish, Cyrillic and Greek.

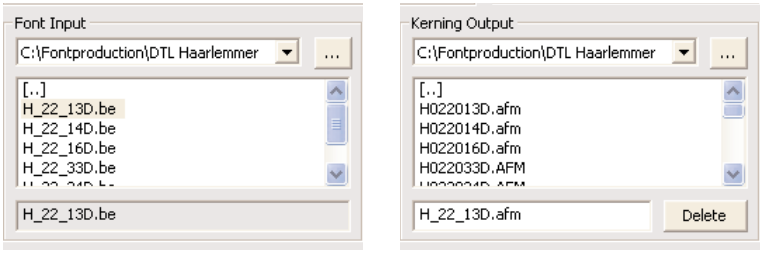
It is standard procedure to generate a kerning file that contains all applicable combinations for a font database. This is not only helpful for OpenType production but also for the generation of single code pages with DTL DataMaster. Automatically only the relevant kerning pairs for this code page will be exported.

If an extensive kerning file is used and the number of kern pairs is

smaller than the total amount of combinations, the combinations with the smallest kerning values will be removed. This process will be repeated until the entered number in the Max. Num of Pairs has been reached.

3. Input/Output control

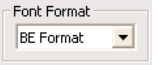
A simple interface is used to control respectively the font input and the resulting kerning output.



It is easy to control the input/output data.

3.1 *Font Input*

Before selecting one or more font databases, the font database format has to be chosen first. There are three options: *BE Format*, *IK Format* and *IK*, *BE*. Because DTL KernMaster can generate kerning in batch, multiple databases can be selected.

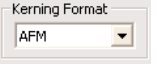


The format options are BE, IK or both.

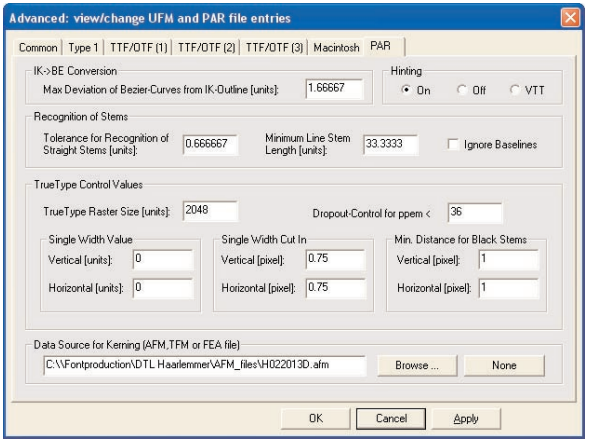
3.2 *Kerning Output*

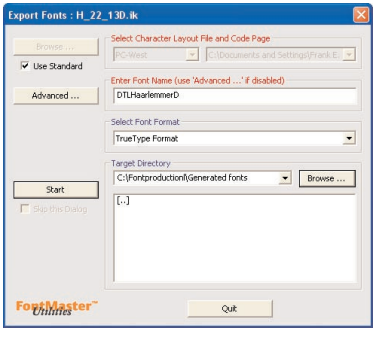
Here a folder/directory has to be chosen for the output. The output format, AFM or FEA, has to be selected first. The name of the resulting Adobe Font Metrics or kern feature file can be changed. The default setting is the name of the selected font database plus the suffix AFM or FEA.

The *Edit UFM* button opens the UFM file that is in the same directory as the font database file. This way it is for instance possible to directly connect the newly generated kerning information in the PAR dialog of the UFM file.



The output format can be either AFM or FEA.





In the PAR section of the UFM file the newly generated kerning file can be connected to the font database.

4. Result Listing

After processing the kerning pairs DTL KernMaster will show a listing which can be printed, saved and viewed. This is especially useful when kerning is generated in batch. If the *View Listing…* option is chosen, the file will be opened in the standard browser.

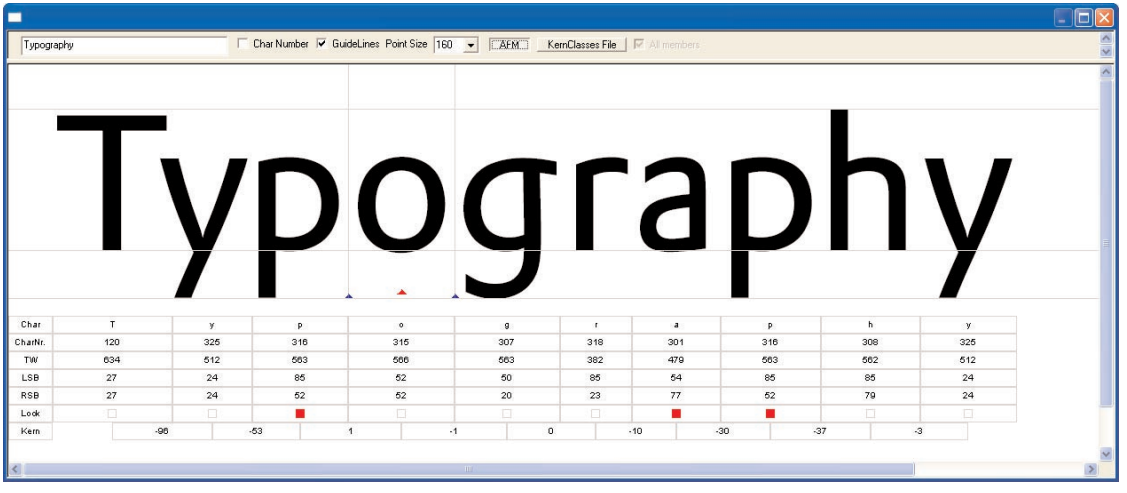
```
DTL KernMaster 2.00.009A : Mar-25-2004 12:00:00
Batch processing is being started.
* kerning class file is "eMac 80GB:Applications (Mac OS 9):Dutch Type
Library:DTL FontMaster 2.00.009A:kernmaster.cla"
* kerning pair file is "eMac 80GB:Applications (Mac OS 9):Dutch Type
Library:DTL FontMaster 2.00.009A:kernmaster.krn"
* character layout file is "eMac 80GB:Applications (Mac OS 9):Dutch Type
Library:DTL FontMaster 2.00.009A:beeditor.cha"
* encoding key is "ANNumLat1"
+ kerning calculation ist being started
+ input: "eMac 80GB:Desktop Folder:AFM_Output:D_19_13T.BE"
+ output: "eMac 80GB:Desktop Folder:AFM_Output:D_19_13T.afm"
- output: "eMac 80GB:Desktop Folder:AFM_Output:D_19_13T.afm"
- input: "eMac 80GB:Desktop Folder:AFM_Output:D_19_13T.BE"
- kerning calculation has been finished
```

The listing of the kerning process can be viewed, printed and saved.

5. Checking the outcome

The output by DTL KernMaster can be checked and edited in the Metrics Editor of both DTL Bezier- and IkarusMaster. Therefore the AFM file or FEA file has to be selected in the Metrics Editor. Any changes made will be saved automatically to the original metrics file.

An AFM or FEA file can be selected in the Metrics Editor of DTL Bezier- and IkarusMaster

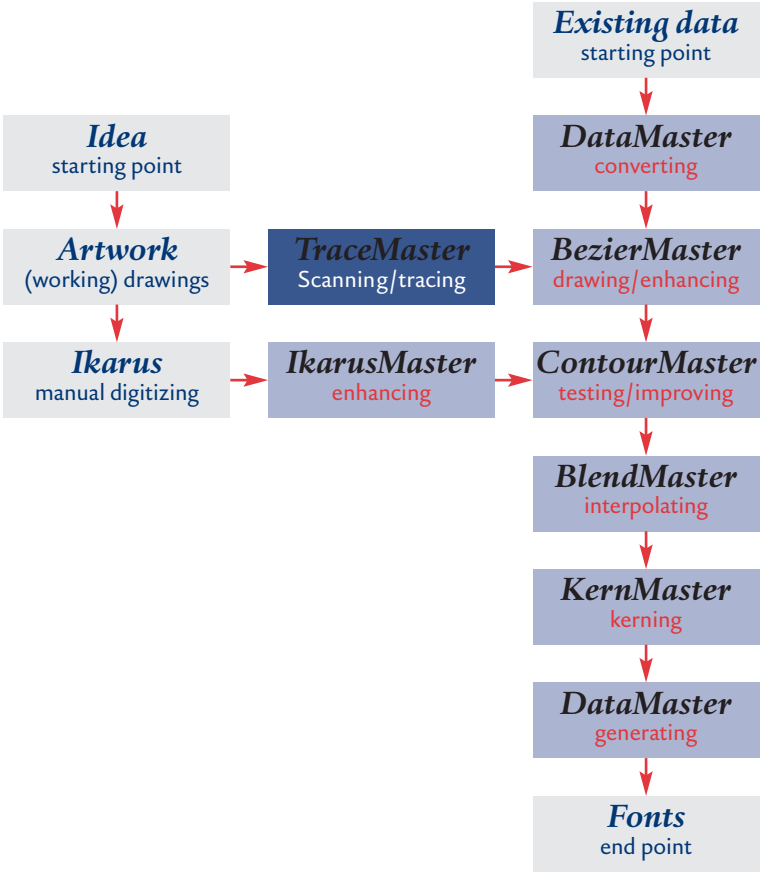


It is possible to check and alter the kerning in the Metrics Editor of both DTL Bezier- and IkarusMaster.

Dutch Type Library

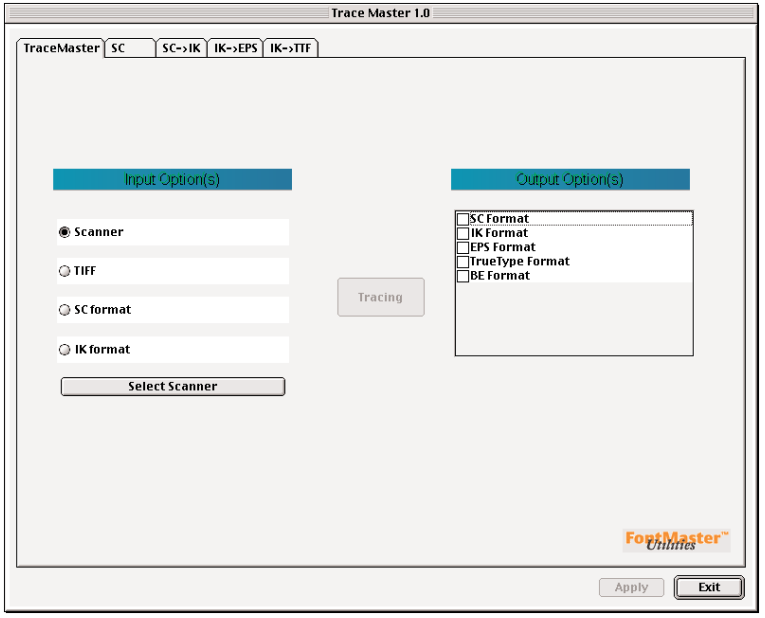
DTL TraceMaster

’s-Hertogenbosch / Hamburg
Summer 2004



The diagram shows a typical workflow based on the modules of DTL FontMaster.

DTL TraceMaster is the easy to handle module for autotracing scanned data. The program is perfectly suited for tracing letters and logo's and the points on the contours will be handled according to the rules for font outlines, which means that for instance extreme points will be placed perfectly. DTL TraceMaster supports scanners directly through the Twain drivers, a standard used by most manufacturers, which makes it possible to scan and directly convert analog data into high quality digital contours. DTL TraceMaster supports several outline description formats, including the BE, IK, EPS formats. It is even possible to convert the scanned data directly into a TrueType font. DTL TraceMaster is mostly used to convert TIFF data into BE format. Because the accuracy is exceptional, DTL TraceMaster is a good alternative for digitizing contours by hand. The generated contours in BE or IK format can be edited in respectively DTL BezierMaster and DTL IkarusMaster. The EPS format can, of course, be imported in Adobe Illustrator or Macromedia Freehand for further processing.



The main dialog shows the Input and Output options.

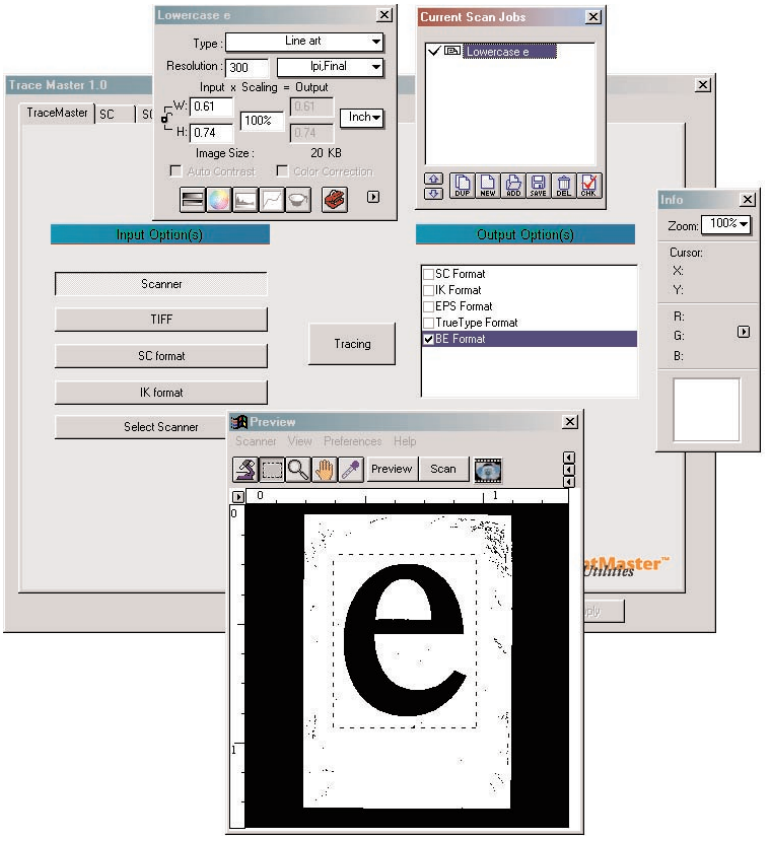
Starting DTL TraceMaster

The output in BE, IK and EPS format is based on an EM-square of 15000 x 15000 units.The relation between the size of the scanned (TIFF) model and the output resolution by DTL TraceMaster is simple: 1 centimetre is converted into 1000 units. The quality of the output depends on the resolution and size of the original model. A good starting point is a caps height of 10 centimetres. The outlines saved in BE, IK or EPS format can be scaled if necessary in respectively DTL BezierMaster, DTL IkarusMaster and Adobe Illustrator, of course.

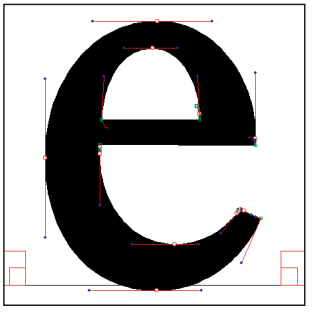
The original TIFF data must be line art. The size of the original model determines the size and resolution of the output; when scanning a range of letters at the same resolution, changing the size of the cut outs (Canvas Size) does not cause scaling by the program. This way the original relation in size of the characters is preserved although the Canvas Sizes differ.

Main dialog

The main dialog has two rows with options:
–*Input Option(s)*
–*Output Option(s)*
Furthermore it makes it possible to select a scanner in case there are more supporting Twain drivers on the system or to switch to the other functions using the tabs on top of the dialog.



Because DTL TraceMaster supports the Twain driver format, images can be scanned directly from the program and automatically be converted into outlines.



The resulting contour generated by DTL TraceMaster after scanning.

1. Input Option(s)

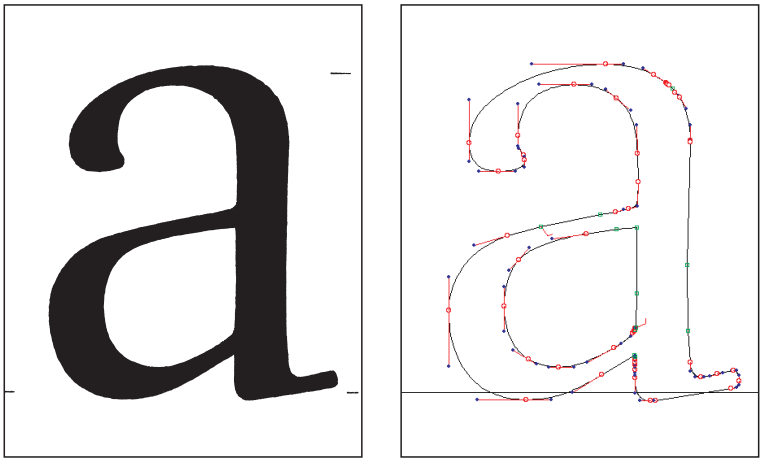
Before an image can be scanned or converted first the source has to be selected.

1.1 *Scanner*

In case the scanner is supported by a Twain driver normally DTL TraceMaster will detect the apparatus automatically. The program selects the driver which has been set up as the default. If more scanners are attached to the computer, the *Select Scanner* button can be used to switch between these.

Scanning is only possible after a format is chosen from the *Output Option(s)* list. The *Tracing* button in the center of the dialog will be activated then and by clicking on it with the mouse the image will be scanned, converted into the chosen format and saved in the selected directory. Please pay attention to the fact that only line art can be handled by DTL TraceMaster. In case the scanner is not Twain compatible, use the sofware that came with the apparatus and save the image as line art in TIFF format. Subsequently use DTL Trace Master.

TIP: In case the scanner is not Twain compatible, use the program that came with your scanner and save the scan as line art in TIFF format. Subsequently use DTL TraceMaster to convert the TIFF image into an outline description.



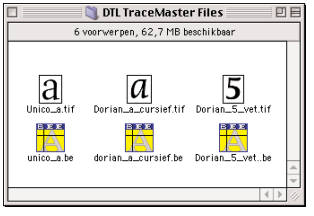
At the left the original TIFF file is shown. It is recommended to remove as much as possible dust and other unwanted stuff from the model. At the right the traced image (without any editing) is shown.

Only line art can be handled by the program.

1.2 TIFF

Images saved as TIFF (Tagged-Image File Format) can be converted into BE, IK, and EPS outlines. The TIFF images must contain line art, otherwise the conversion will fail. The generated outlines will be saved automatically in the same directory in which the TIFF file is located. The accuracy of the output depends on the size of the image. A good starting point is a caps height of 10 centimetres.

It is recommended to remove as much as possible dust and other unwanted stuff from the original model because of the accuracy every details will be traced by the program.



The generated outlines will be automatically saved in the directory in which the original TIFF files are located.

1.3 SC Format

The SC format is a bitmap format developed by URW++. Normally you will not use this but at the other hand the SC format is one of the output options of DTL TraceMaster. The SC format can be placed in the background of DTL BezierMaster and DTL IkarusMaster.

1.4 IK Format

The Ikarus format can also be directly converted into the formats listed in the *Output Option(s)*: EPS, TrueType and BE. In case EPS is selected and the IK data contains multiple characters, by default only the first character will be converted into EPS format. The generated EPS outline will be scaled by default to an EM-square of 15000 x 15000 units. The default settings can be altered in the dialog that shows up when the IK->EPS tab on top of the main dialog is selected.

If BE is selected as output format, all characters in the IK font will always be converted.

2. Output Option(s)

Depending on the selected *Input Option(s)*, several output formats will be shown.

2.1 SC Format

The SC format is a bitmap format developed by URW++. Normally you will not use this but at the other hand the SC format is one of the output options of DTL TraceMaster. The SC format can be placed in the background of DTL BezierMaster and DTL IkarusMaster

2.2 IK Format

The Ikarus format can also be directly converted into the formats listed in the *Output Option(s)*: EPS, TrueType and BE. In case EPS is selected and the IK data contains multiple characters, by default only the first character will be converted into EPS format. The generated EPS outline will be scaled by default to an EM-square of 15000 x 15000 units. The default settings can be altered in the dialog that shows up when the IK->EPS tab on top of the main dialog is selected.

If BE is selected as output format, all characters in the IK font will always be converted.

2.3 EPS Format

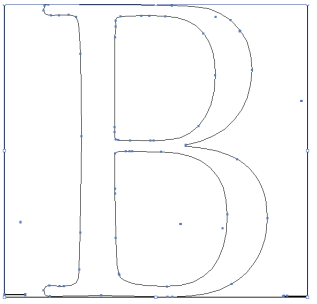
The Encapsulated PostScript format is a widely used file format developed by Adobe Systems. It describes a document written in the PostScript language and it contains all the code necessary to print the file. EPS files can be edited using Adobe Illustrator or Macromedia Freehand. EPS files can be imported in DTL BezierMaster.

2.4 TrueType Format

This is a very versatile font format jointly developed by Apple and Microsoft. It is the default font format for the Mac OS and Windows operating systems. The standard EM-square of a TrueType font measures 2048 x 2048 units.

2.5 BE Format

This is the bezier description format developed by URW++ and used internally by DTL BezierMaster. The actual description of the contours is at the end the same as in a genuine PostScript font.

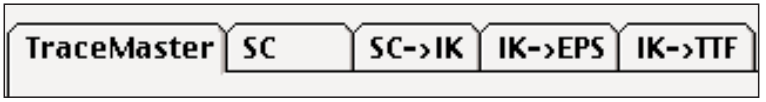


Images can be saved in EPS format directly.

After the conversion is done succesfully, a confirmation dialog appears.

3. Select Scanner

In case more Twain drivers are supported by the system, this option makes it possible to chose a particular scanner.

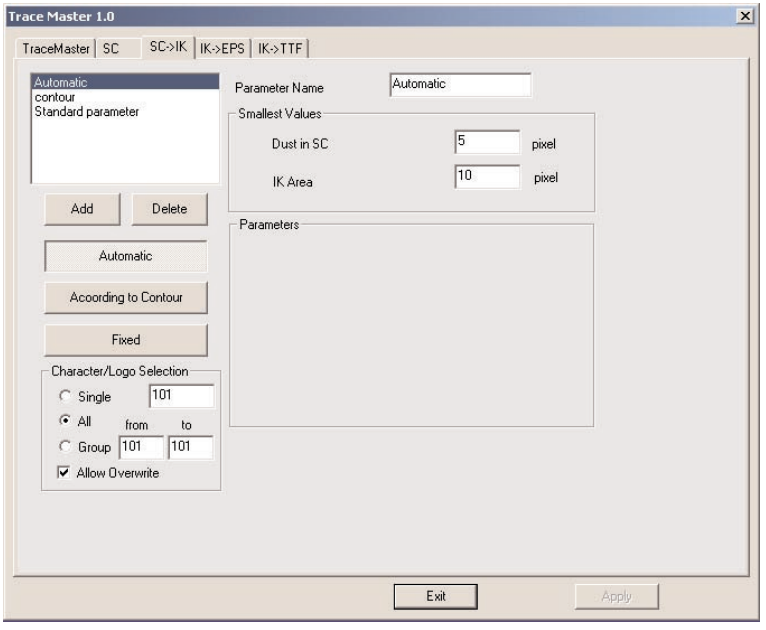


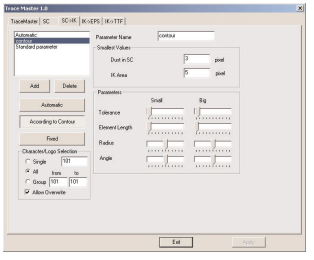
4. Tab Options

On top of the main dialog there are several tab options to chose from depending on the selected output format.

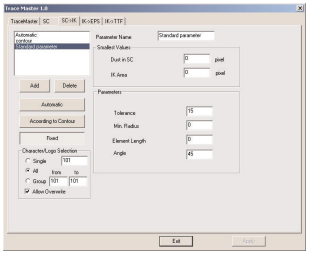
4.1 *SC*

This dialog makes it possible to give the resulting SC file a so called *SC Logo Number* and to delete the original TIFF that was used as model.





The three SC->IK dialogs.



4.2 *SC->IK*

There are three dialogs for parametrization of the conversion from the bitmap format into the IK outline: *Automatic*, *Contour* and *Standard Parameter*.

–*Automatic*

This is the default setting; DTL TraceMaster will automatically determine

the tolerance for the conversion. The default for *Dust in* SC is five pixels and for IK *Area* ten pixels. These values can be altered; smaller values will result in the conversion of more particles. Larger values will remove more dust.

–*Contour*

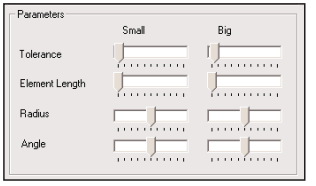
This dialog will show some more parameters besides *Dust in* SC and IK *Area*.

With *Tolerance* the deviation from the original is fixed. In case the quality of the original is low it is recommended to enlarge the tolerance to prevent very rough outlines containing a lot of IK points.

With *Elementh Length* the way the contour is described is determined. Small values will result in contours containing short straight lines. Larger values will result in curves. It is for this reason recommended to use high values for the *Elementh Length* parameter.

A contour can exist of straight lines and curves. With *Radius* the tolerance between corners and curves is defined; higher values will replace the curves by corners. Therefore it is recommended not to change the default value, which is 'medium'.

With *Angle* the number of curves can be influenced. The default is also 'medium'. Basically this value should not be altered.



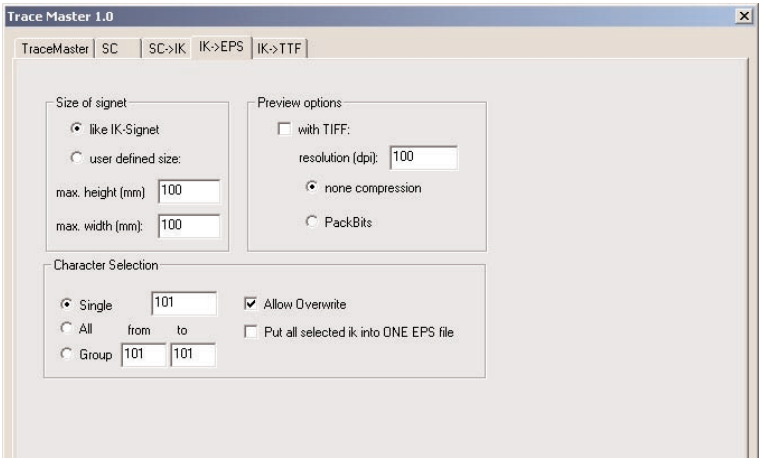
In case the option Contour is chosen from the SC->IK *dialog, several parameters can be altered.*

–*Standard Parameter*

Here it is possible for the user to define his own personal settings for all parameters described under *Contour*. These preferences will be automatically preserved when the program is closed and can be used for future projects. This is especially very useful for large projects that are for instance handled by different people.

4.3 *IK->EPS*

In this dialog the parameters for the conversion of the IK data into the EPS outline can be controlled.



The IK->EPS *dialog.*

–*Size of signet*

Here the relation between the size of the original IK data and the EPS that has to be generated can be defined. The default for IK data is an EM-square of 15000 x 15000 units and 1000 units are equal to ten millimetres. The option *like IK-Signet* will preserve the original size of the Ikarus outline. With *user define size* the output size of the EPS can be defined.

–*Preview options*

The newly generated EPS file can contain a TIFF file for preview purposes. The resolution of this TIFF can be defined. There is also an option to compress the TIFF file. The default is *none compression.*

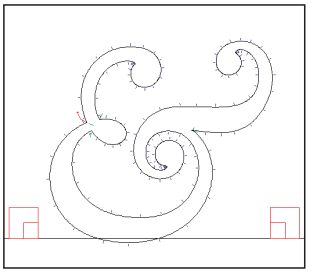
–*Character Selection*

An IK database file can contain multiple glyphs, especially when the database consists of a font. In case *Single* is selected, the Character Number must be entered; the default is number 101, which is normally the capital A in a font database.

The option *All* will convert all characters in the IK database into EPS format.

The option *Group* will convert all characters within the defined range.

Allow overwrite will overwrite a single EPS file with the same number or a range of EPS files depending of the above described options. *Put all selected IK into one EPS file* will generate a large EPS containing all characters.

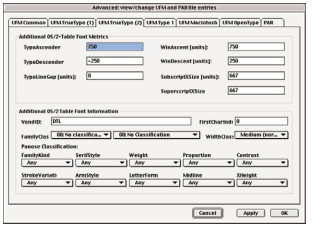


Ikarus data can be converted directly into EPS format.

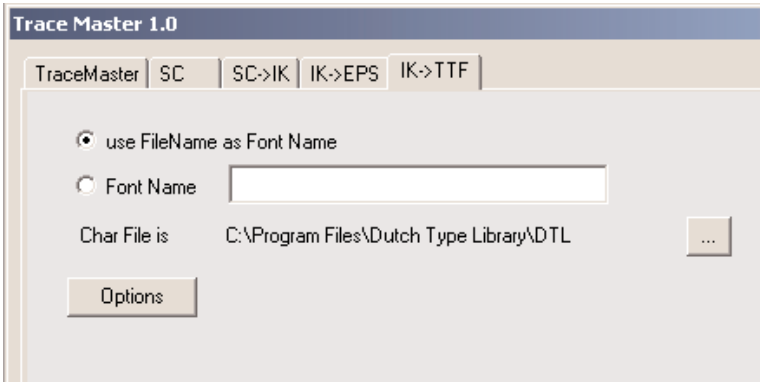
4.4 *IK->TTF*

It is possible to use DTL TraceMaster to generate TrueType fonts directly from IK data. In the main dialog the *Font Name* can be entered. By default the Font Name is taken from the File Name.

The *Options* button will reveal a large range of parameters for the TrueType font. Please check the DTL DataMaster manual for a detailed description of this functionality.



For a detailed description of all the parameters that can de defined for a TrueType font, the DTL DataMaster manual can be checked.




Dutch Type Library
FontMaster™
Utilities

Dutch Type Library

Appendices

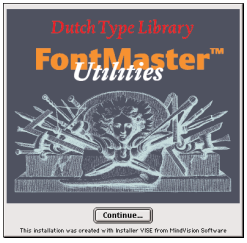
's-Hertogenbosch / Hamburg
Summer 2004

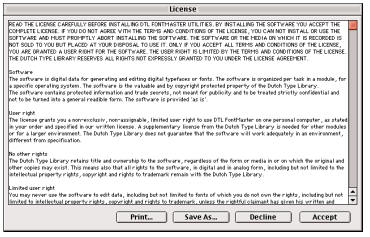
1. Using the Mac OS installer

System requirements: Mac OS 8.6 or higher.
Physical RAM: 64 MB (96 MB or more recommended).
Screen resolution: 800 x 600 or higher.
Free hard disk space: 40 MB.

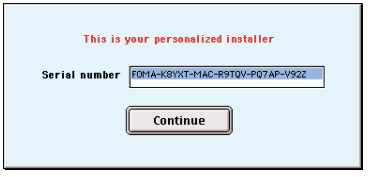
1.1 *Installing DTL FontMaster*

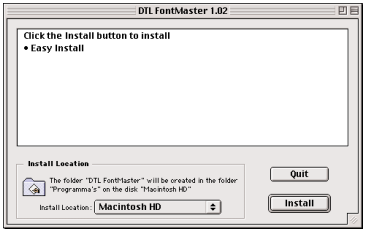
Double click on the installer icon. The installer will start with a splash screen. Click on the *Continue …* button.





You will be notified to read the *License Agreement* (see Appendix II). In case you accept, your personalized serial number will be shown.



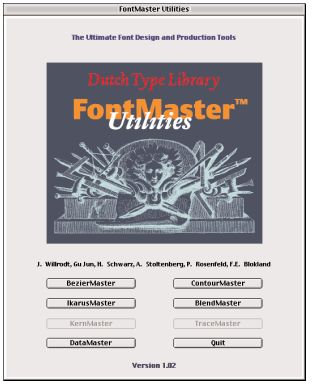


After continuing a new dialog will appear and you have to select the folder where DTL FontMaster will be installed. After clicking the *Install* button the program will unpacked and copied to your hard disk.

By default an alias will be placed in the **Apple** menu. You can easily remove DTL FontMaster by dragging the folder that contains the program files, Microsoft libraries, etcetera to the dustbin.

It is also possible to install the DTL FontMaster modules separately. Therefore the DTL FontMaster BASE file has to be installed first. This contains the Microsoft libraries, the DLL's, etcetera. An up to date BASE file can always be downloaded from the DTL FontMaster web site (http://www.fontmaster.nl). Subsequently the module has to be installed. Take care to select the folder where the BASE file has been installed as the destination folder.

After a succesfull installation DTL FontMaster can be activated from the Apple menu. Grey buttons indicate that the module belonging to the button is not installed.



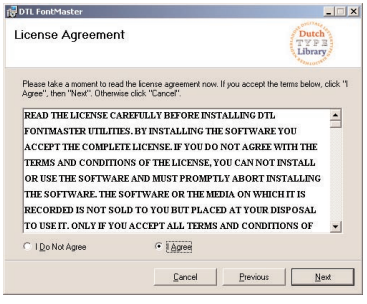
2. Using the Windows installer

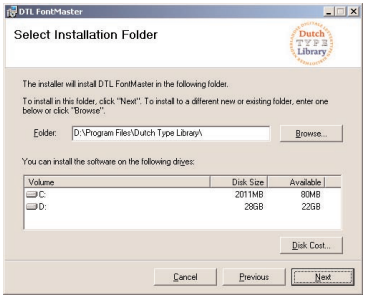
System requirements: Windows: 95/98, ME, NT, 2000, XP.
Physical RAM: 64 MB (96 MB or more recommended).
Screen resolution: SVGA or higher.
Free hard disk space: 25 MB.

2.1 *Installing DTL FontMaster*

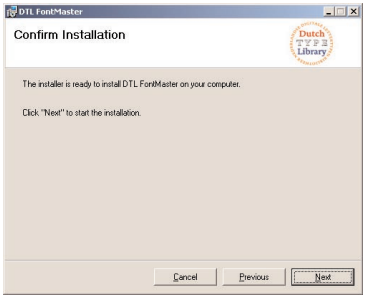
Double click on the installer icon. The installer will start with a splash screen. Click on the *Next* button.

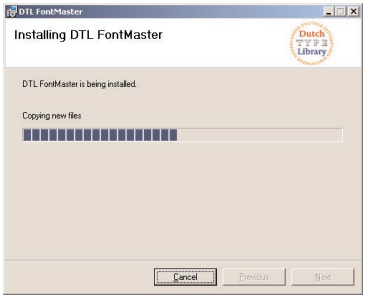
A welcome dialog will be shown; click on the *Next* button.





You will be notified to read the *License Agreement* (see Appendix II). In case you agree the *Next* button will be activated, otherwise you have to cancel the installation. In case you agree, a new dialog will appear and you have to select the folder where DTL FontMaster will be installed. After clicking the *Next* button, you must confirm your choice.

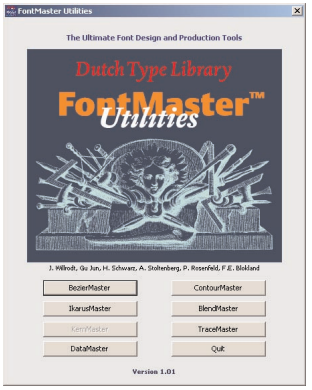




The default directory for installing DTL FontMaster is C:\Program Files\ Dutch Type Library\. After DTL FontMaster has been installed, you will find the program in the Windows **Start** menu under Programs–> Dutch Type Library -> DTL FontMaster. It is not necessary to restart your computer.

In case DTL FontMaster has to be repared or removed, the installer can be used for these purposes also. When you try to install DTL FontMaster on a system where already a copy of the program has been installed, you will be notified to remove the installed copy first.

It is also possible to install the DTL FontMaster modules separately. Therefore the DTL FontMaster BASE file has to be installed first. This contains the Microsoft libraries, the DLL's, etcetera. An up to date BASE file can always be downloaded from the DTL FontMaster web site (http://www.fontmaster.nl). Subsequently the module has to be installed. Take care to select the folder where the BASE file has been installed as the destination folder. The default directory for the installation of the FM modules is C:\Program Files\ Dutch Type Library\. In case the installation was succesfull the appropriate button in the 'central switch board' will be activated.



After a succesfull installation DTL FontMaster can be activated from the Windows Start menu. Grey buttons indicate that the module belonging to the button is not installed.

Appendix II: End User License

Read the license carefully before installing *DTL FontMaster Utilities*. By installing the software you accept the complete license. If you do not agree with the terms and conditions of the license, you can not install or use the software and must promptly abort installing the software. The software or the media on which it is recorded is not sold to you but placed at your disposal to use it. Only if you accept all terms and conditions of the license, you are granted a user right for the software. The user right is limited by the terms and conditions of the license. The Dutch Type Library reserves all rights not expressly granted to you under the license agreement.

Software

The software is digital data for generating and editing digital typefaces or fonts. The software is organized per task in a module, for a specific operating system. The software is the valuable and by copyright protected property of the Dutch Type Library.

The software contains protected information and trade secrets, not meant for publicity and to be treated strictly confidential and not to be turned into a general readible form. The software is provided 'as is'.

User right

The license grants you a non-exclusiv, non-assignable, limited user right to use DTL FontMaster on one personal computer, as stated in your order and specified in our written license. A supplementary license from the Dutch Type Library is needed for other modules or for a larger environment. The Dutch Type Library does not guarantee that the software will work adequately in an environment, different from specification.

No other rights

The Dutch Type Library retains title and ownership to the software, regardless of the form or media in or on which the original and other copies may exist. This means also that all rights to the software, in digital and in analog form, including but not limited to the intellectual property rights, copyright and rights to trademark remain with the Dutch Type Library.

Limited user right

You may never use the software to edit data, including but not limited to fonts of which you do not own the rights, including but not limited to intellectual property rights, copyright and rights to trademark, unless the rightful claimant has given his written and signed consent.

Proprietary rights and obligations

The software is the valuable and by copyright protected property of the

Dutch Type Library. You will not make or have made, or permit to be made, any copies of the software, documentation, or any portions thereof, except one (= 1) copy solely for backup purposes. Any such copy of the software shall contain the same proprietary notice which appears on or in the original software. You agree not to distribute the software, nor to sell, lend or transfer it from one computer to another via a network. You agree not to change the software.

Term

The license is effective until terminated. The Dutch Type Library has the right to terminate the license immediately if you fail to comply with any term of the license. In addition, the Dutch Type Library reserves the right to claim punitive damages. Upon such termination you will destroy the original and any copies of the software and related documentation and cease all use of the trademarks. You may terminate the license any time by destroying the original and any copies of the software and related documentation.

Limited warranty

The Dutch Type Library warrants the physical media upon which the software was furnished to you to be free of defects in workmanship and manufacture for a period of four (= 4) weeks from the date of delivery to you as evidenced by a copy of your receipt. If failure of any physical media has resulted from accident, abuse, or misapplication, the Dutch Type Library shall have no responsibility to replace the physical media. The Dutch Type Library does not and cannot warrant the performance or results you may obtain by using the software or documentation. The Dutch Type Library does not make any warrants of any kind, either expressed or implied, including, but not limited to the implied warranty of fitness for a particular purpose.

Limit of liability

In no event will the Dutch Type Library be liable to you for consequential or incidential damage, including damage from loss of business profits or savings, business interruption, loss of business information, and the like, or for claim by any party arising out of the use of or inability to use the software, even if the Dutch Type Library has been advised of the possibility of such damage.

Payment

You are granted a user right for the software, on condition that you pay the invoice that goes with the license within the term of payment. In case the invoice is not paid in time, the user right is terminated immediately and any use of the software to and from that moment is automatically illegal. After

expiration of the term of payment, the Dutch Type Library has the right to charge interest and collecting-costs and to demand the return of the software and any copies.

Indemnification

You agree to indemnify and hold the Dutch Type Library harmless from and against any claims or damage which may result from your breach of this license agreement.

Governing law

The license will be governed by the laws in force in the Netherlands.

You acknowledge that you have read the license, understand it and that it is the complete and exclusive statement of your agreement with the Dutch Type Library which supersedes any prior agreement, oral and written, and any other communications between the Dutch Type Library and you relating to the subject matter of the license, and that your obligations under this agreement shall inure to the benefit of Dutch Type Library licensors whose rights are licensed under this agreement. No variation of the terms of this agreement will be enforceable against the Dutch Type Library unless the Dutch Type Library gives it expressed consent in writing signed by an officer of the Dutch Type Library. By installing the software you accept your own liability to comply with all terms and conditions of the license. If you do not agree completely with the license, promptly abort the installation of the software.

http://www.dutchtypelibrary.com

Appendix III: Character Layout Files

This document describes the format of the Character Layout Files used in DTL FontMaster for conversion of the BE and IK databases into the PostScript Type 1, TrueType and OpenType formats and vice versa. The basic function of these layout files is to renumber the characters from one layout to another one.

1. General structure

–The Character Layout File is an editable ASCII file.
–All Character Layout Files have the extension '**.cha**' appended to the file name.
–Each line in this file can have a length of up to 255 characters.
–Lines starting with a lowercase '**c**' or capital '**C**' are ignored by the DTL FontMaster modules and can be used for comments.
–A special line with the keyword 'Version' is used to denote the version of the format specification (this is not a version number for the file itself). This format specification defines Version 002.000.
–Keywords are case insensitive and can consist out of maximal 16 characters concisting the lowercase letters a–z, the capitals A–Z and the digits 0–9.
–Empty lines in the file are allowed.
–Tabs and Spaces are allowed in all lines and are ignored.
–The conversion information from one numbering system to another one is contained in several columns. The content of each column is characterized by a keyword.
–The entries for the different columns are separated by '**;**'.
–The character information is included in two lines with the keywords '**Starttable**' and '**Endtable**'.
–The keywords for the content of the column have to follow the line with the keyword '**Starttable**'.

NOTE: The version number of the format specicification must be 002.000 in all cases, otherwise the Character Layout File will be not accepted by the FM modules. When the user wants to attach a version number to a proprietary .cha file, this information should be preceded by a 'C' (comment).

2. Proprietary Character Layout Files

Characters are stored by number in a BE and IK database. The database number corresponds with a PostScript name and a Unicode number via the Character Layout Files. By default four .cha files come with DTL FontMaster: beeditor.cha, TTBAS.cha, urwotf.cha, winuni.cha.

It is of course possible that FM users create proprietary encoding systems. If necessary they can use PostScript names that are located in the 'private area' in combination with Private Use Area Unicode codepoints. Normally these characters will be placed at positions in the database that are undefined. This automatically means that also a proprietary Character Layout File has to be created because the default .cha files will not 'recognize' the database numbers and therefore the characters will not be exported when a font is created.

In case a font with a proprietary codepage containing PostScript names and Unicodes in the 'private area' is imported and converted by DTL DataMaster into a BE or IK database, the characters will be automatically placed in undefined character cells. Please note that the import conversion is directly influenced by the selected Character Layout File.

2.1 *Structure*

The structure of a Character Layout File is basically quite simple. The sequence of columns in the .cha file is arbitrary and a column is identified by name. Keywords have to be known to the program in order to know the datatype of the column, i.e integer for QDNum, strings for PSName, etc. A Character Layout File can contain one or more codepages. The beeditor.cha file for instance contains multiple codepages for Western and Eastern European, Greek, Cyrillic, etc. codepages for Mac OS and Windows.

The relation between the keyword and the names of the codepages shown in the Font Administration tool and by DTL DataMaster is hardcoded. It is not possible to define proprietary keywords because these will not be recognized by the programs. It is recommended to use the fixed keywords QDNum and ANNum and UNINum for customized layouts.

2.2 *Keywords*

The list of hardcoded keywords in the Font Administration tool in DTL BezierMaster and DTL IkarusMaster is:
UNINum;URWNum;UNINumUp; Unicode and URW Number, UNINumUp (UpperCase) is not used
ANNumLat1;QDNumLat1; Western European (PC/Mac OS)
ANNumLat2;QDNumLat2; Eastern European (PC/Mac OS)
ANNumGr;QDNumGr; Greek (PC/Mac OS)
ANNumTu;QDNumTu; Turkish (PC/Mac OS)
ANNumCy;QDNumCy; Cyrillic (PC/Mac OS)
ANNumHe;QDNumHe; Hebrew (PC/Mac OS)
ANNumBa; Baltic (PC)
QDNumRo; Romanian (Mac OS)
ANNumSym;QDNumSym; Symbols (PC/Mac OS)
ANNumKazahk;QDNumKazakh; Kazakh (PC/Mac OS)
ANNumUzbek;QDNumUzbek; Uzbek (PC/Mac OS)
PSName

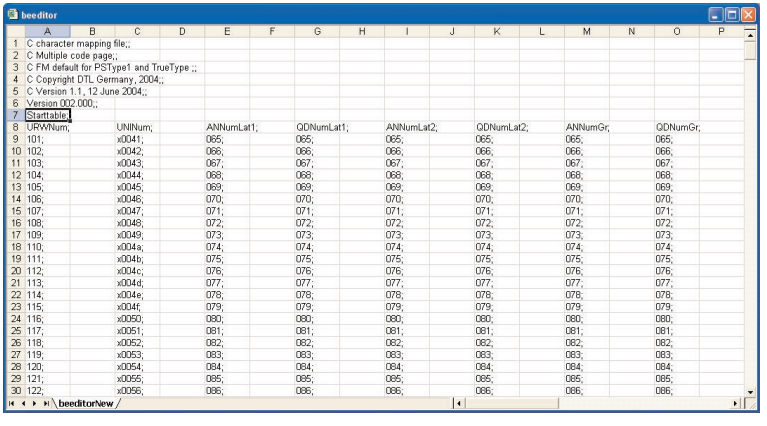
The next page shows more allowed keywords and some extra information about the above listed hardcoded keywords. Other keywords are also possible and will be defined if requested. Please note that not all listed keywords are currently supported by DTL FontMaster.

Keyword	*Interpretation*	*Datatype*	*Range*
URWnum	Number in the URW++ Layout	long	1–65534
URWcomp	URW++ Number for components of composite characters	long	1–65534
PSNum	PostScript Number (for Standard Encoding)	short	1–255
PSName	PostScript Name	charstring	128 ASCII
QDNum	QuickDraw Number (Mac Layout)	short	1–255
ANNum	ANSI Number (Windows)	short	1–255
HPNum	HP Master Symbol List	long	0–65535
IFNum	Intellifont Character Plane Number	long	0–65535
CGNum	Compugraphic Character Code	long	0–65535
ISONum	Number in ISO-Standard Code	long	
UNINum	Number in Unicode	long	0–65535
CharClass	Classification of Character (Upper or Lower Case, digit)	long	
StaClass	Weight of character for calculation of stemsnaps and statistical evaluation	long	
KernClass	Classification of character to calculate kerning values	long	M,V,LR,RL, etcetera
JISNum	JIS Number for Kanji characters	long	0–65535
JEFNum	Fujitsu Numbering (extended JIS)	long	0–65535
KUTENNum	Kuten Numbering for Kanji	long	0–65535
KenjiBangoNum	Fujitsu Database Numbering	long	0–65535
GlyInd	Index in TrueType font	long	0–65535
GBNum	GB Number	long	0–65535
SJISNum	Shift JIS Number	long	0–65535
MSGB	GB Number for Microsoft	long	0–65535
MSB5	B5 Number for Microsoft	long	0–65535
MACGB	GB Number for Mac	long	0–65535
MACB5	B5 Number for Mac	long	0–65535
URWVNum	Character Number for vertical writing	long	0–65535

```
C character mapping file;;
C Multiple code page;;
C FM default for PSType1 and TrueType ;;
C Copyright DTL Germany, 2004;;
C Version 1.1, 12 June 2004;;
Version 002.000;;
Starttable;;
URWNum;UNINum;ANNumLat1;QDNumLat1;ANNumLat2;QDNumLat2;ANNumGr;QDNumGr;ANNumTu;QDNumTu;ANNumCy;QDNumCy;ANNumHe;
QDNumHe;ANNumBa;QDNumRo;ANNumSym;QDNumSym;ANNumKazahk;QDNumKazahk;ANNumUzbek;QDNumUzbek;PSName
101;x0041;065;065;065;065;065;065;065;065;065;065;065;065;065;065;065;65;65;65;65;A
102;x0042;066;066;066;066;066;066;066;066;066;066;066;066;066;066;066;66;66;66;66;B
103;x0043;067;067;067;067;067;067;067;067;067;067;067;067;067;067;067;67;67;67;67;C
104;x0044;068;068;068;068;068;068;068;068;068;068;068;068;068;068;068;68;68;68;68;D
105;x0045;069;069;069;069;069;069;069;069;069;069;069;069;069;069;069;69;69;69;69;E
106;x0046;070;070;070;070;070;070;070;070;070;070;070;070;070;070;070;70;70;70;70;F
107;x0047;071;071;071;071;071;071;071;071;071;071;071;071;071;071;071;71;71;71;71;G
108;x0048;072;072;072;072;072;072;072;072;072;072;072;072;072;072;072;72;72;72;72;H
109;x0049;073;073;073;073;073;073;073;073;073;073;073;073;073;073;073;73;73;73;73;I
110;x004a;074;074;074;074;074;074;074;074;074;074;074;074;074;074;074;74;74;74;74;J
111;x004b;075;075;075;075;075;075;075;075;075;075;075;075;075;075;075;75;75;75;75;K
112;x004c;076;076;076;076;076;076;076;076;076;076;076;076;076;076;076;76;76;76;76;L
113;x004d;077;077;077;077;077;077;077;077;077;077;077;077;077;077;077;77;77;77;77;M
114;x004e;078;078;078;078;078;078;078;078;078;078;078;078;078;078;078;78;78;78;78;N
115;x004f;079;079;079;079;079;079;079;079;079;079;079;079;079;079;079;79;79;79;79;O
116;x0050;080;080;080;080;080;080;080;080;080;080;080;080;080;080;080;80;80;80;80;P
117;x0051;081;081;081;081;081;081;081;081;081;081;081;081;081;081;081;81;81;81;81;Q
118;x0052;082;082;082;082;082;082;082;082;082;082;082;082;082;082;082;82;82;82;82;R
119;x0053;083;083;083;083;083;083;083;083;083;083;083;083;083;083;083;83;83;83;83;S
120;x0054;084;084;084;084;084;084;084;084;084;084;084;084;084;084;084;84;84;84;84;T
121;x0055;085;085;085;085;085;085;085;085;085;085;085;085;085;085;085;85;85;85;85;U
122;x0056;086;086;086;086;086;086;086;086;086;086;086;086;086;086;086;86;86;86;86;V
123;x0057;087;087;087;087;087;087;087;087;087;087;087;087;087;087;087;87;87;87;87;W
```

Part of a Character Layout File.

To make editing of the Character Layout Files easy, Microsoft Excell files are used. These are included in the installer and can be found in the 'Dutch Type Library' directory.



Microsoft Excel document.

Appendix IV: UFM File Format

Keywords

In the following table the keywords are listed with the type of the entry and their influence in the production.

II	= code in II-FINF section
Type:	i = numerical value
	s = text
	s2 = Japanese twobyte (SJIS) text
RC (on Mac)	generation of FOND, SFNT or NFNT resource
+	entry is interpreted by DTL FontMaster module
–	entry is ignored by DTL FontMaster module

	Program DM			*Conversions to*					
Keyword	**II**	**Typ**	**Calculation**	**UFM**	**TI**	**AFM**	**PFM**	**TT**	**RC**
AccentOffset	28	i	Y-min of 701 minus Y-min of 751	+	–	–	–	–	–
Ascender	16	i	304 / Header	+	–	+	+	–	–
AscenderHHEA	265	i	Default [13]	+	–	–	–	+	–
Bodysize	71	i	Header	+	+	+	+	+	–
CapHeight	30	i	Header / 108	+	+	+	+	–	–
Comment	17	s	Input by hand	–	+	+	–	–	–
Copyright	18	s	Input by hand / Default [2]	+	+	+	+	+	–
Descender	29	i	316 / Header	+	–	+	+	–	–
DescenderHHEA	266	i	Default [4]	+	–	–	–	+	–
DoubleLower-UnderlineOffset	138	i	Y-max of lower stroke of 1155	+	–	–	+	–	–
DoubleLower-UnderlineWidth	140	i	Y-thickness of lower stroke of 1155	+	–	–	+	–	–
DoubleUpper-UnderlineOffset	137	i	Y-max of 1155	+	–	–	+	–	–
DoubleUpper-UnderlineWidth	139	i	Y-thickness of upper stroke of 1155	+	–	–	+	–	–
ExternalLeading	144	i	Default [7]	*	–	–	+	–	–
FaceName	128	s	input by hand	*	–	–	+	–	+
FamilyClass	257	i	Default [3]	+	–	–	–	+	–
FamilyName	19	s	input by hand	*	+	+	–	–	–
FigureSize	34	i	Y-max of 510 minus baseline undershoot of 510	+	+	–	–	–	–
FirstCharIndex	281	i	input by hand / Default first Unicode character in font	+	–	–	–	+	–
FondAscender	68	i	Default [1]	+	–	–	–	–	+
FondDescender	70	i	Default [4]	+	–	–	–	–	+
FondID	64	i	input by hand / Listing	*	–	–	–	–	+
FondLeading	69	i	20% of Bodysize (SBS)	+	–	–	–	–	+

	Program DM			*Conversions to*					
Keyword	**II**	**Typ**	**Calculation**	**UFM**	**TI**	**AFM**	**PFM**	**TT**	**RC**
FondName	65	s	input by hand	*	–	–	–	–	+
FontFamilyName	269	s	input by hand	*	–	–	–	+	–
FontName	20	s	input by hand	*	+	+	+	+	+
FsType	277	i	input by hand / Default [6]	+	–	–	–	+	–
FullName	21	s	input by hand	*	+	+	–	–	–
Identifier	32	s	input by hand	–	+	–	–	–	–
InternalLeading	143	i	Default [7]	+	–	–	+	–	–
IsFixedPitch	22	s	input by hand	*	+	+	+	+	–
JPNCopyright	271	s2	input by hand	–	–	–	–	+	–
JPNFontFamilyName	273	s2	input by hand	–	–	–	–	+	–
JPNSubfamilyName	274	s2	input by hand	–	–	–	–	+	–
JPNTrueTypeID	272	s2	input by hand	–	–	–	–	+	–
JPNVersion	275	s2	input by hand	–	–	–	–	+	–
LineGap HHEA	267	i	Default [7]	+	–	–	–	+	–
LowestRecPpem	276	i	input by hand / Default [6]	+	–	–	–	+	–
MacFileName	66	s	input by hand	*	–	–	–	–	+
MacStyle	67	i	input by hand	*	–	–	+	+	+
Notice	23	s	input by hand / Default [2]	+	+	+	–	–	–
Panose	258	s	Default [5]	+	–	–	–	+	–
PCWeight	129	i	input by hand	*	–	–	+	+	–
PitchAndFamily	130	s	Default [6]	+	–	–	+	–	–
Slant	145	i	Header	+	+	–	+	–	–
StrikeOutOffset	141	i	Y-min of 1426	+	–	–	+	+	–
StrikeOutWidth	142	i	Y-thickness of 1426	+	–	–	+	+	–
SubfamilyName	270	s	input by hand	*	–	–	–	+	–
SubScript	132	i	Default [7]	+	–	–	+	+	–
SubScriptSize	134	i	Y-max of 596 minus baseline undershoot of 596	+	–	–	+	+	–
SubScriptxSize	263	i	Width of 596	+	–	–	–	+	–
SuperScript	131	i	Y-min of 584 minus baseline undershoot of 596	+	–	–	+	+	–
SuperScriptSize	133	i	Y-max of 596 minus baseline undershoot of 596	+	–	–	+	+	–
SuperScriptxSize	264	i	Width of 584	+	–	–	–	+	–
TrueTypeID	268	s	Default [8]	+	–	–	–	+	–
TypoAscender	261	i	Default [12]	+	–	–	–	+	–
TypeDescender	262	i	Default [4]	+	–	–	–	+	–
TypoLineGap	260	i	Default [7]	+	–	–	–	+	–
UnderlineOffset	135	i	Y-max of 1154	+	–	–	+	–	–
UnderlinePosition	24	i	Y-center of 1154	+	+	+	–	+	–
UnderlineThickness	25	i	Y-thickness of 1154	+	+	+	–	+	–
UnderlineWidth	136	i	Y-min of 1154 minus Y-max of 1154	+	–	–	+	–	–

	Program DM			*Conversions to*					
Keyword	**II**	**Typ**	**Calculation**	**UFM**	**TI**	**AFM**	**PFM**	**TT**	**RC**
UniqueID	33	i	input by hand	*	+	–	–	–	+
VendID	259	s	input by hand / Default [9]	+	–	–	–	+	–
Version	26	s	input by hand / Default [10]	+	+	+	–	+	–
Weight	27	s	input by hand	*	+	+	–	–	–
WidthClass	256	i	Default [11]	+	–	–	–	+	–
WinAscent	278	i	input by hand, IK units / Default Y-max ANSI	+	–	–	–	+	–
WinDescent	279	i	input by hand, IK units / Default Y-min ANSI	+	–	–	–	+	–
xavgCharWidth	280	i	input by hand, IK units / Default average width	+	–	–	–	+	–
xHeight	31	i	Header / 326	+	+	+	+	–	–

1 Header: Cap height (CAH)
2 URW Software, Copyright YEAR by URW
3 0 0
4 Header: Distance baseline - lower body line (BAL)
5 0 0 0 0 0 0 0 0 0 0
6 Dontcare
7 0
8 VendID: FontName: YEAR
9 URW
10 001.005
11 5
12 12000
13 Bodysize (Header, SBS) * 1.2 minus Descender (Header, BAL)

Other font vendors should be aware that the defaults 2, 8, 9 and 10 may not match their production. Some font depending defaults like 4, 5, 6 and 11 may be altered by hand.

Example of an UFM file

```
StartFontMetrics
/UniqueID get 5060740 eq exch/FontType get 1 eq and}{pop false}ifelse
{save true}{false}ifelse}{false}ifelse
20 dict begin
/FontInfo 16 dict dup begin
  /version (001.000
TTName 9 3 1 0x409 "Frank E. Blokland"; #Windows
TTName 9 1 0 0 "Frank E. Blokland"; #Macintosh
TTName 11 3 1 0x409 "http://www.dutchtypelibrary.com"; #Windows
TTName 11 1 0 0 "http://www.fontmaster.nl"; #Macintosh
TTName 14 3 1 0x409 "http://www.dutchtypelibrary.com"; #Windows
TTName 14 1 0 0 "http://www.fontmaster.nl"; #Macintosh
Copyright Dutch Type Library, 2001. All rights reserved
Notice Generated on 08-04-2001/  File# D019C16T
Version 002.00E
FamilyName DTLDocumentaST
FontName DTLDocumentaST-Bold
FullName DTL Documenta ST Bold
UniqueID 5060740
Weight Bold
IsFixedPitch false
Ascender 766
Descender -234
UnderlinePosition -133
UnderlineThickness 20
Bodysize 1000
CapHeight 683
FigureSize 515
XHeight 486
AccentOffset 133
MacFileName DTLDocSTBol
FondName DTL Documenta ST Bold
FondID 13740
MacStyle 1
FondAscender 766
FondDescender -234
FondLeading 0
FaceName DTL Documenta ST
PCWeight 6
PitchAndFamily Dontcare
SubScript 0
SubScriptSize 600
SubscriptXSize 667
SuperScript 400
SuperScriptSize 600
SuperscriptXSize 667
UnderlineOffset -123
UnderlineWidth 20
DoubleUpperUnderlineOffset 106
DoubleUpperUnderlineWidth 53
DoubleLowerUnderlineOffset -340
DoubleLowerUnderlineWidth 67
StrikeOutOffset 267
StrikeOutWidth 53
InternalLeading 0
ExternalLeading 0
```

```
Slant 0
FontFamilyName DTLDocumentaST
SubfamilyName Bold
TrueTypeID D019C16T
WidthClass 5
FamilyClass 0 0
Panose 0 0 0 0 0 0 0 0 0 0
VendID DTL
TypoAscender 766
TypoDescender -234
TypoLineGap 0
AscenderHHEA 950
DescenderHHEA -250
LineGapHHEA 0
WinAscent 766
WinDescent 234
FirstCharIndex 0
EndFontMetrics
```

The following pages show the usage of the keywords in the UFM files.

Ascender

Format	AFM
Structure	Global font information
Element	Ascender
Comment	Top of lower case d

Format	PFM
Structure	EXTTEXTMETRIC
Element	etmLowerCaseAscent
Comment	Distance that the ascender of lowercase letters extends above the baseline

Ascender HHEA

Format	TTF
Structure	'hhea' table
Element	Ascender
Comment	Distance from baseline of highest ascender, typographic Ascent (Apple)

Bodysize

Format	All
Structure	None
Element	None
Comment	Used to scale the UFM entries to the target grid

CapHeight

Format	AFM
Structure	Global font information
Element	CapHeight
Comment	Top of upper case H

Format	PFM
Structure	EXTTEXTMETRIC
Element	etmCapHeight
Comment	The height of uppercase characters

Format	PFM
Structure	PFMHEADER
Element	dfAscent, CapHeight, dfInternalLeading
Comment	Specifies the distance from the top of a character definition cell to the baseline of the typographical font. It is useful for aligning the baseline of fonts of different height
Format	TI
Structure	Private dictionary
Element	BlueValues
Comment	Cap-height alignment

Comment

Format	AFM

Format TI
Structure Private dictionary
Element BlueValues
Comment Cap-height alignment

Format AFM **Comment**
Structure Global font information
Element Comment
Comment Arbitrary text, may be present in an AFM file

Format AFM **Copyright**
Structure Global font information
Element Comment
Comment Text regarding the copyright

Format PFM
Structure PFMHEADER
Element dfCopyright(60)
Comment as AFM

Format TI
Structure FontInfo dictionary
Element Copyright
Comment as AFM

Format TTF
Structure 'name' table
Element Name ID 0
Comment as AFM

Format AFM **Descender**
Structure Global font information
Element Descender
Comment Bottom of lower case p

Format PFM
Structure EXTTEXTMETRIC
Element etmLowerCaseDescent
Comment Distance of lower case descender extending below baseline

Format TTF **Descender HHEA**
Structure 'hhea' table
Element Decender
Comment Distance from baseline of lowest descender, typographic descent (Apple)

DoubleLowerUnderline-Offset

Format PFM
Structure EXTTEXTMETRIC
Element etmDoubleLowerUnderlineOffset
Comment Offset downward from the baseline where the top of the lower double underline should appear

DoubleLowerUnderline-Width

Format PFM
Structure EXTTEXTMETRIC
Element etmDoubleLowerUnderlineWidth
Comment Thickness of the lower double underline bar

DoubleUpperUnderline-Offset

Format PFM
Structure EXTTEXTMETRIC
Element etmDoubleUpperUnderlineOffset
Comment Offset downward from the baseline where the top of the upper double underline should appear

DoubleUpperUnderline-Width

Format PFM
Structure EXTTEXTMETRIC
Element etmDoubleUpperUnderlineWidth
Comment Thickness of the upper double underline bar

ExternalLeading

Format PFM
Structure PFMHEADER
Element dfExternalLeading
Comment Amount of extra leading that the designer requests the application to add between rows

FaceName

Format PFM
Structure PFMHEADER
Element dfFace(pointer)
Comment Microsoft Windows font Name. Up to four fonts may build up a family having the same FaceName. These fonts must make up two pairs with the same dfWeight, but different to the entry of the other pair. The two fonts having the same dfWeight must differ in the dfItalic byte

Format Mac font suitcase
Structure FOND resource
Element Resource Name
Comment Name of the FOND seen in the menu if the resource does contain a family of several fonts

FamilyClass

Format TTF
Structure 'OS2' table
Element sFamilyClass
Comment First number in UFM describes IBM Font-Family class, the second one the IBM Subfamily class. This parameter is intended for use in selecting an alternate font when the present one is not available

FamilyName

Format AFM
Structure Global font information
Element FamilyName
Comment Name of the 'font family' to which the font belongs

Format TI
Structure FontInfo dictionary
Element FamilyName
Comment Human-readable name for a group of fonts that are stylistic variants of the same design. All fonts that are members of such a group should have exactly the same FamilyName. It should be suitable for use in a font selection menu

FigureSize

Format TI
Structure Private dictionary
Element BlueValues
Comment figure-size alignment

FirstCharIndex

Format TTF
Structure 'OS2' table
Element usFirstCharIndex
Comment Minimum Unicode index (char.code) in this font according to the cmap subtable for platform ID 3 and encoding ID 0 or 1. Should be 0x0020 for most fonts supporting Win-ANSI or other char.sets

FondAscender

Format Mac font suitcase
Structure FOND resource
Element ffAscent
Comment Makes up the baseline-to-baseline distance together with FondDescender and FondLeading on a Mac

FondDescender

Format Mac font suitcase
Structure FOND resource
Element ffDescent
Comment Makes up the baseline-to-baseline distance together with FondAscender and FondLeading on a Mac

Format	Mac font suitcase	**FondID**
Structure	FOND resource	
Element	ffFamID	
Comment	FOND Family ID	
Format	Mac font suitcase	
Structure	NFNT resource	
Element	Resource ID	
Comment	Together with the UniqueID a NFNT Resource ID is calculated	
Format	Mac font suitcase	
Structure	SFNT resource	
Element	Resource ID	
Comment	Together with the UniqueID a SFNT Resource ID is calculated	
Format	Mac font suitcase	**FondLeading**
Structure	FOND resource	
Element	ffLeading	
Comment	Makes up the baseline-to-baseline distance together with FondAscender and FondDescender on a Mac. Some applications assume this value to be equal to zero	
Format	Mac font suitcase	**FondName**
Structure	FOND resource	
Element	Resource Name	
Comment	Name of the FOND seen in the menu if the resource contains only one font	
Format	TTF	**FontFamilyName**
Structure	'name' table	
Element	NameID 1	
Comment	The name the user sees in the font-menu	
Format	TTF	
Structure	'name' table	
Element	NameID 4	
Comment	First part of FullFontName	
Format	AFM	**FontName**
Structure	Global font information	
Element	Font Name	
Comment	Name of the font program as presented to the PostScript language *findfont* operator	

FontName

Format	TI
Structure	FontInfo dictionary
Element	FontName
Comment	Font's name, passed to the PostScript *define* font operator by program TO. Should be unique. Can be a condensation of the FullName by removing spaces. It is customary to limit it's length to less than 40 characters.

Format	TTF
Structure	'name' table
Element	NameID 6
Comment	PostScript name for the font separated to base- and suffix names. Each suffix starts with an uppercase letter after an hyphen has occured in the Name

FullName

Format	AFM
Structure	Global font information
Element	FullName
Comment	Full text name of the font

Format	TI
Structure	FontInfo dictionary
Element	FullName
Comment	Unique, human-readable name for an individual font. Typically, it begins with the FamilyName and continues with various style descriptors separated by spaces

FsType

Format	TTF
Structure	'OS2' table
Element	fsType
Comment	Indicates font embedding licensing rights for the font. Makes temporary loading of a font possible by an embedding-aware application. This licensing rights are granted by the vendor of the font

Internal Leading

Format	PFM
Structure	PFMHEADER
Element	dfinternalLeading
Comment	Amount of leading inside the bounds set by the dfPixHeight member. Accent marks may occur in this area

Format	PFM
Structure	PFMHEADER
Element	dfAscent, CapHeight, dfInternalLeading
Comment	see **CapHeight**

IsFixedPitch

Format AFM
Structure Writing direction metrics; mono/proportionally
Element IsFixedPitch
Comment If true, this indicates that the font program is a monospaced font. A value false indicates a proportionally spaced font

Format TI
Structure FontInfor dictionary
Element IsFixedPitch
Comment As AFM

Format PFM
Structure PFMHEADER
Element dfPitchandFamily
Comment As AFM. If true, the low order 4 bits are set to 0x00, else 0x01

Format TTF
Structure 'post' table
Element IsFixedPitch
Comment As AFM. False is interpreted as 0, true is interpreted as 1

LineGapHHEA

Format TTF
Structure 'hhea' table
Element LineGap
Comment Typographic line gap (Apple)

LowestRecPpem

Format TTF
Structure 'head' table
Element LowestRecPPEM
Comment Smallest readable size in pixels

MacFileName

Format Mac font suitcase
Structure Mac file system
Element Filename of the suitcase: FileName.scr or FileName.tt
Comment Filename seen in the folder if FOND contains NFNT

MacStyle

Format Mac font suitcase
Structure FOND resource
Element Font Style
Comment Determines the style under which the font appears in the menu. It is:
0 Regular or single font
1 Bold in the family
2 Italic in the family
3 Bold Italic in the family

Format PFM
Structure PFMHEADER
Element dfItalic
Comment If MacStyle equals 2 or 3, dfItalic is set to 1 so that the font is recognized as Italic

Format TTF
Structure 'head' table
Element macStyle
Comment as FOND

Format TTF
Structure 'OS2' table
Element fsSelection
Comment MacStyle 0 –> fsSelection 64
MacStyle 1 –> fsSelection 32
MacStyle 2 –> fsSelection 1
MacStyle 3 –> fsSelection 33

Notice

Format AFM
Structure Global font information
Element Notice
Comment Font name trademark or copyright notice

Format TI
Structure FontInfo dictionary
Element As AFM
Comment As AFM

Panose

Format TTF
Structure 'OS2' table
Element Panose
Comment 10 numbers to describe the visual characteristics of a font

PCWeight

Format PFM
Structure PFMHEADER
Element dfWeight
Comment Weight of the characters on a scale from 1–1000. The value of the UFM is taken times 100. It means:
1 thin
2 extra light
3 light
4 normal
5 medium

Format 6 semi bold
Structure 7 bold
Element 8 extra bold
Comment 9 heavy

Format TTF
Structure 'OS2' table
Element usWeightClass
Comment As PFM

Format PFM **PitchAndFamily**
Structure PFMHEADER
Element dfPitchAndFamily
Comment Indicates, in a general way, the look of a font. There are:
– Dontcare
– Roman
– Swiss
– Modern
– Script
– Decorative

Format AFM
Structure Global font information
Element ItalicAngle
Comment Angle in degrees counter-clockwise from the vertical of dominat vertical strokes of the font

Format PFM
Structure EXTTEXTMETRIC
Element etmSlant = –Slant
Comment Angle in in tenth of degrees clockwise from the upright version of the font

Format TI
Structure FontInfo dictionairy
Element As AFM
Comment As AFM

Format PFM **StrikeOutOffset**
Structure EXTTEXTMETRIC
Element etmStrikeOutOffset = StrikeOutOffset + StrikeOutWidth
Comment Offset upward from the baseline where the top of a strike-out bar should appear

Format TTF
Structure 'OS2' table
Element yStrikeoutPosition
Comment Position of the bottom of the strike-out relative to the baseline

StrikeOutWidth

Format PFM
Structure EXTTEXTMETRIC
Element etmStrikeOutWidth
Comment Thickness of the strike-out bar

Format TTF
Structure 'OS2' table
Element yStrikeoutSize
Comment as PFM

SubfamilyName

Format TTF
Structure 'name' table
Element NameID 2
Comment Address only style and weight

Format TTF
Structure 'name' table
Element NameID 4
Comment If not equal to Regular, second part of FullFontName

SubScript

Format PFM
Structure EXTTEXTMETRIC
Element etmSubScript
Comment Recommend vertical offset of subscript characters from baseline

Format TTF
Structure 'OS2' table
Element ySubscriptYOffset
Comment As PFM

SubScriptSize

Format PFM
Structure EXTTEXTMETRIC
Element etmSubScriptSize
Comment Recommend vertical size of subscript characters

Format TTF
Structure 'OS2' table

Element ySubscriptYSize
Comment As PFM

Format TTF **SubscriptXSize**
Structure 'OS2' table
Element ySubscriptXoffset
Comment Recommended horizontal size of subscript characters

PFM
Format EXTTEXTMETRIC
Structure etmSuperScript
Element Recommended vertical offset of superscript characters from
Comment the baseline

Format TTF
Structure 'OS2' table
Element ySuperscripttYoffset
Comment As PFM

Format PFM **SuperScriptSize**
Structure EXTTEXTMETRIC
Element etmSuperScriptSize
Comment Recommended vertical size of superscript characters

Format TTF
Structure 'OS2' table
Element ySuperscripttYSize
Comment As PFM

Format TTF **SuperScriptXSize**
Structure 'OS2' table
Element ySuperscripttXSize
Comment Recommended horizontal size of superscript characters

Format TTF **TrueTypeID**
Structure 'name' table
Element NameID 3
Comment Unique identifier that applications can store to identify the
font beeing used

Format TTF **TypoAscender**
Structure 'OS2' table
Element sTypoAscender
Comment New typographic ascender. One good source is the Ascender

TypoDescender

Format TTF
Structure 'OS2' table
Element sTypoDescender
Comment New typographic descender. One good source is the Descender value from an AFM file

TypoLineGap

Format TTF
Structure 'OS2' table
Element sTypoLineGap
Comment New typographic line gap. Typical values average 7–10% of units per em

UnderlineOffset

Format PFM
Structure EXTTEXTMETRIC
Element etmUnderlineOffset
Comment Offset downward from the baseline where the top of a single bar should appear

UnderlinePosition

Format AFM
Structure Global font information
Element UnderlinePosition
Comment Recommended distance from the baseline for centering underlining strokes. This is the y coordinate of the center of the stroke

Format TI
Structure FontInfo dictionairy
Element Underline position
Comment As AFM

Format TTF
Structure 'post' table
Element Underline Position
Comment As AFM

UnderlineThickness

Format AFM
Structure writing direction metrics
Element UnderlineThickness
Comment Recommend stroke width for underlining

Format TI
Structure FontInfo dictionary
Element UnderlineThickness
Comment As AFM

Format	TTF
Structure	'post' table
Element	UnderlineThickness
Comment	As AFM

UnderlineWidth

Format	PFM
Structure	EXTTEXTMETRIC
Element	etmUnderlineWidth
Comment	Thickness of the underline bar

UniqueID

Format	TI
Structure	Private dictionary
Element	UniqueID
Comment	Integer in the range from 0 to 16777215 that uniquely identifies the font. The numbers from 4000000 to 4999999 form an open range and may be used in a controlled environment. To distribute a font widely a UniqueID should be obtained from Adobe Systems Inc.

Format	Mac font suitcase
Structure	NFNT Resource
Element	Resource ID
Comment	Togethet with the FondID a nfnt Resource ID is calculated

VendID

Format	TTF
Structure	'OS2' table
Element	achVendID(4)
Comment	Four character identifier for vendor of given font

Version

Format	AFM
Structure	Global font information
Element	Version
Comment	Font program version identifier

Format	TI
Structure	FontInfo dictionairy
Element	version
Comment	As AFM

Format	TTF
Structure	'head' table
Element	fontRevision
Comment	Version number

Format TTF
Structure 'name' table
Element NameID 5
Comment Release and version information from the font vendor

Weight

Format AFM
Structure Global font information
Element Weight
Comment Weight of the font. E.g. Bold

Format TI
Structure FontInfo dictionary
Element Weight
Comment Human readable name for the weight or 'boldness' attribute of a font

WidthClass

Format TTF
Structure 'OS2' table
Element usWidthClass
Comment Relative change from the normal width to height ratio by a fontdesigner for the glyphs in a font. It means:
1 Ultra-Condensed
2 Extra-Condensed
3 Condensed
4 Semi-Condensed
5 Medium (normal)
6 Semi-Expanded
7 Expanded
8 Extra-Expanded
9 Ultra-Expanded

WinAscent

Format TTF
Structure 'OS2' table
Element usWinAscent
Comment Ascender metric for windows, yMax for all characters in the Windows ANSI set. For platform 3 encoding 0 fonts same as yMax

WinDescent

Format TTF
Structure 'OS2' table
Element usWinDescent
Comment Descender metric for Windows, –yMin for all characters in the Windows ANSI set. For platform 3 encoding 0 fonts same as –yMin

XavrCharWidth

Format TTF
Structure 'OS2' table
Element xAvgCharWidth
Comment Average of the width of all of the 26 lowercase letters a through z. If any of the 26 lowercase letters are not present, this parameter should equal the weighted average of all glyphs. For non-UGL (platform 3, encoding 0) fonts, use the unweighted average.

XHeight

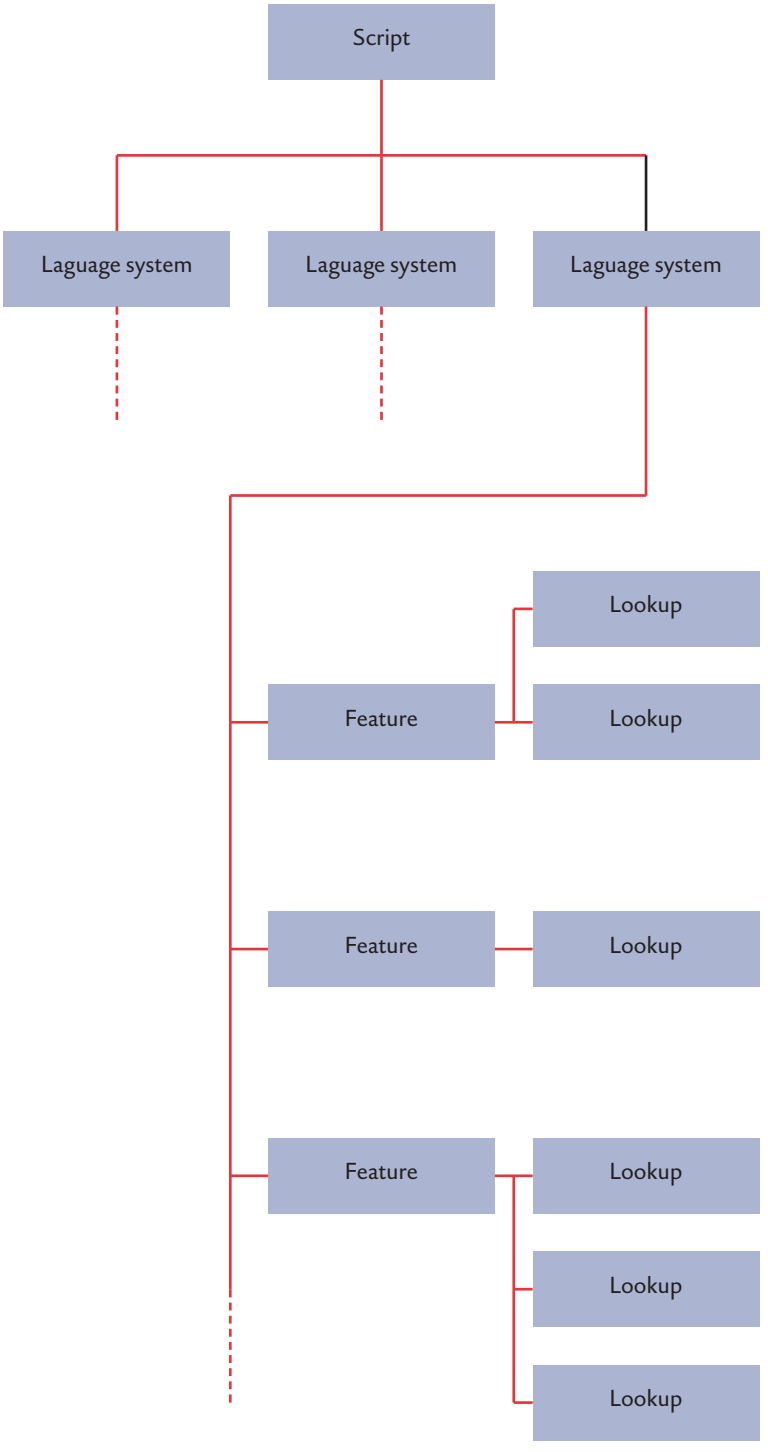
Format AFM
Structure Global font information
Element XHeight
Comment Top of lower case x

Format PFM
Structure EXTTEXTMETRIC
Element etmXHeight
Comment Height of lower case letters in the font

Format TI
Structure Private dictionary
Element BlueValues
Comment x-height alignment

The OpenType Layout Model

OpenType layout data is organized by script, language system, typographic feature and lookup.



Contents

1. A short history of font technology

Before 1980 Proprietary and hardware dependent font formats (bitmap, vector).

1974–1978 Ikarus outline font format (open format, machine independent, font data base format).

Mid 1980s Scalable font formats (outline + hints).
– URW vs, BS.
– Type 1 (based on Bezier and URW-like hints).
– F3, Bitstreams Speedo and others …

Late 1980s Development of TrueType by Apple (Unicode based, instructions, flexible and expandable).
– Implementation on the Macintosh in 1990.
– Implementation in Windows 3.1 in 1991.

1991 Opening of Type 1 Format (Adobe) (1- Byte font format).
To font format for 2-Byte fonts.

1993 CID font format for CJK (2-Byte).
– Took 5–6 years to appear on the market.

1994 TrueType GX (advanced Layout features).
– Failed on the market.

1995 TTO (multilingual support, Layout features for Arabic).
TTC (TrueType Collection Files).

1996 SFNT-Wrapped CID Fonts (Adobe, Mac platform).

1997 OpenType specification.

Increasing complexity

1.1 *Conclusion*

– Font technology has been rapidly developed during the last 20 years.
– Font technology has become a very important part in the computerized world.
– Parallel to globalization, fonts have been been extended to complex scripts like Arabic, Indic, Thai etc. and to large character sets for China, Japan and Korea.
– Fonts are becoming more and more complex, which puts more pressure on the font developer and designer.
– The evolution of the font formats also allows the use of fine typographic features.

2. What is OpenType ?

Open Type is more than a simple font format, it is an architecture with building blocks:
– OpenType fonts.
– Operating System support.
– Application support.
– Printer support.

OpenType fonts have four essential ingredients:
– Outline description (Bezier, quadratic splines …).
– Hinting information for screen optimization (hints, instructions).
– Character mapping tables.
– Features (for glyph substitution and positioning).

OpenType fonts come in two flavours:
– Type 1 outlines, hints (.otf)
– TrueType outlines, instructions (.ttf)

There is no standard as to what an OpenType font must contain (this might be difficult for the customer and but also for marketing):
– 256 – >50000 glyphs.
– hundreds of features or none.

2.1 *OS Support*

OpenType fonts should work on different platforms (Windows, Mac OS, Linux). Windows 2000 and XP support both OTF flavours natively and support many features (not all) through its Uniscribe API and the OTLS (OpenType Layout Services Library). Mac OS 9.2 and OS X support for both OTF flavours is limited. Glyph access and rendering is supported but there is no OS support for layout features. Apple supports instead its own Apple Advanced Technology (AAT) technology, which is a renamed version of GX. This means that fonts which should work on both platforms must support both OpenType layout tables as well as the AAT tables. Linux should support OpenType through Freetype.

2.2 *Applications*

Applications are using the outlines, hints and feature tables. Adobe has implemented the feature font support into the applications such as InDesign, PhotoShop, etcetera. These programs are platform independent, and OS independent).

3. The structure of OpenType fonts

OpenType fonts have a common table structure like TTFs (also called SFNT on the Macintosh). OpenType Fonts may use Type 1-like outlines and hints or TrueType-like outlines and hints. The reason for that was probably that neither Microsoft nor Adobe wanted to throw away the considerable amount of work which had been done on the Type 1 and TrueType architectures.

Advantages of Type 1-like outlines (CFF table):
– Simple hinting structure, intelligence in the rasterizer.
– Thousands of existing Type 1 fonts can be converted without quality loss.
– Bezier outlines are familiar to (type) designers.

Advantages of TrueType outlines (GLYF table):
– Powerful instructions for suberb screen quality.
– Quadratic spline outlines.

Other information is stored in common tables, such as:
– cmap for the mapping of glyphs –> Unicode code points.
– head, hhea for header information.
– OS/2 for general font information.
– Gasp for greyscaling.

Essential for OpenType are the following tables:
– GPOS *glyph positioning*
– GSUB *glyph substitution*
– GDEF *glyph definition*
– BASE *baseline table for different scripts*
– JSTF *justification*
– DSIG *digital signature*

The main difference with simple TrueType fonts is the presence of some of the above listed tables which allow access to glyphs which have no direct Unicode codepoint. For complex scripts, i.e. writing systems that require some degree of character reordering and/or glyph processing to display, print or edit text (such as Arabic or Indic) Open Type tables are absolutely necessary.

Using this technology permits the font developer to implement:
– OpenType Layout fonts allow a rich mapping between characters and glyphs, which supports ligatures, positional forms, alternates, and other substitutions.
– OpenType Layout fonts include information to support features for two-dimensional positioning and glyph attachment.
– OpenType Layout fonts contain explicit script and language information,

	TrueType (TTF)	Apples TTF (AAT/GX)	OpenType (TTF)	OpenType (OTF)	SFNT-CID (Adobe)
Required	head, hhea, hmtx name OS/2 maxp post cmap	head, hhea, hmtx name OS/2 maxp post cmap	head, hhea, hmtx name OS/2 maxp post cmap **DSIG**	head, hhea, hmtx name OS/2 maxp post cmap	cmap name post
Outline	glyf, loca cvt, fpgm, prep	glyf, loca cvt, fpgm, prep	glyf, loca cvt, fpgm, prep	CFF	CID
Optional	gasp hdmx kern LTSH PCLT VDMX vhea vmtx	gasp hdmx kern vhea vmtx	gasp hdmx kern LTSH PCLT VDMX vhea vmtx	gasp kern vhea vmtx VORG	
Bitmap	EBDT EBLC EBSC	bdat bloc	EBDT EBLC EBSC		bdat bloc
OTF			BASE (baseline data) GDEF (glyph definition) GPOS (glyph positioning) GSUB (glyph substitution) JSTF (Justification)	BASE (baseline data) GDEF (glyph definition) GPOS (glyph positioning) GSUB (glyph substitution) JSTF (Justification)	
AAT		mort, feat, bsln, prop opdb, trak, just … fvar, gvar, Zapf …			faet mort
Adobe					ALMX BBOX FNAM, HFMX, VFMX

The TrueType Font File (Apple's specification AAT)

acnt	*accent attachment table*	hhea	*horizontal header table*
avar	*axis variation table*	hmtx	*horizontal metrics table*
bdat	*bitmap data table*	**hsty**	*horizontal style table*
bhed	*bitmap font header table*	**just**	*justification table*
bloc	*bitmap location table*	kern	*kerning table*
bsln	*baseline table*	**lcar**	*ligature caret table*
cmap	*character code mapping table*	loca	*glyph location table*
		maxp	*maximum profile table*
cvar	*CVT variation table*	**mort**	*metamorphosis table*
cvt	*control value table*	**morx**	*extended metamorphosis table*
EBSC	*embedded bitmap scaling control table*	name	*name table*
fdsc	*font descriptor table*	opbd	*optical bounds table*
feat	*layout feature table*	OS/2	*compatibility table*
fmtx	*font metrics table*	post	*glyph name PostScript compatibility table*
fpgm	*font program table*		
fvar	*font variation table*	prep	*control value program table*
gasp	*gridfitting and scanconversion procedure table*	**prop**	*properties table*
glyf	*glyph outline table*	**trak**	*tracking table*
gvar	*glyph variation table*	vhea	*vertical header table*
hdmx	*horizontal device metrics table*	vmtx	*vertical metrics table*
		Zapf	*glyph reference table*
head	*font header table*		

so a textprocessing application can adjust its behavior accordingly.
– OpenType Layout fonts have an open format that allows font developers to define their own typographical features.

4. The OpenType Layout model

4.1 *Scripts*

Scripts are defined at the top level. A script is a collection of glyphs used to represent one or more languages in writing. For instance, a single script-Latin is used to write English, French, German, and many other languages. In contrast, three scripts –Hiragana, Katakana, and Kanji– are used to write Japanese. With OpenType Layout, multiple scripts may be supported by a single font.

4.2 *Language system*

A language system may modify the functions or appearance of glyphs in a script to represent a particular language. For example, the eszet ligature

Latin
ABCDEFGHIJKLMNabcdefgijk

Cyrillic
АБВГДЕЖЗИЙКЛМабвгдежзи

Greek
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞαβγδεζηθι

CJ (K)
噐噩嚱囑囕嘍呆沓劼势兝勉冰澿

Hangul (K)
가갿곽괡곰극김꿯꼛꼟꿓꼼꿰껠

Hangul Jamo
ᄀ ᄈ ᄋ ᄍ ᄏ ᄒ ᄛ ᄢ ᄲ ᅂ ᅦ ᄫ ᅁ ᅆ

Katakana
あいうえおかきくけこさしすせ

Arabic
ھٹڵگڮڜڤجپێۋۆمکقغطض

Devanagari
अआइईउऊऋऌऍऎएऐऑओओऔकखघङ

FontMaster™ Utilities

OpenType fonts with CFF outlines and AAT support tables.

```
****************************************
*********** Table Directory ************
****************************************
version:      20308.33
numTables:        22
searchRange:     256
entrySelector:     4
rangeShift:       96

tag      offset      length       checksum
----------------------------------------
BASE        364         456       6962C672
CFF         820     6720412       D234DEBC
DSIG   10240852        5788       EADEC4BC
EBDT    6721232     1636487        32BDCD3
EBLC    8357720       67148       883E371E
GPOS    8424868       14600       DD21703D
GSUB    8439468      185706       7F930AE3
OS/ 2   8625176          96       3814B65D
VORG    8625272         812        2BE8ACA
Zapf    8626084      442236       2736C019
cmap    9068320      276664       E31BA3BF
feat    9344984         340       81CD4A53
head    9345324          54       D3061EC9
hhea    9345380          36        8B5416B
hmtx    9345416       72546       D255AEAD
maxp    9417964           6       4F485000
morx    9417972      739840        496DB24
name   10157812        5060       3F369656
post   10162872          32       FFB80032
prop   10162904        3758       DA5761FF
vhea   10166664          36        74F5311
vmtx   10166700       74152       8EFBA4CC
```

is used in the German language system, but not in French or English. And the Arabic script contains different glyphs for writing the Farsi and Urdu languages. In the absence of language- specific rules, default language system features apply to the entire script.

Another example is the hani script which supports China, Korea and Japan. Here we have different glyphs for the same Unicode codepoint for different language systems as can be seen for example in the MS Arial Unicode font:
Script Tag: hani
Language Tag: ZHT, ZHS, KOR

辯 *Chinese traditional*
辩 *Chinese simplified*
辯 *Japanese*

FontMaster™ Utilities

4.3 *Features*

A language system defines features, which are typographic rules for using glyphs to represent a language. The typographic features define the functionality of an OpenType Layout font and are registered in the *OpenType Layout tag registry* at the Microsoft Typography homepage. Font developers can use these features, as well as create their own (if they find an application which uses them!)

Some examples of typographic features are:

– **vert**

This substitutes vertical glyphs in Japanes.

(漢字のテスト。)

(漢字のテスト。)

The substitution of vertical glyphs in Japanese (MS Mincho).

– **init, medi, fina**

A language system feature for the Arabic script substitutes initial, medial, and final glyph forms based on a glyph's position in a word.

Standalone 'ha'

Initial 'ha'

Medial 'ha'

Final 'ha'

– **liga**

Feature for using ligatures in place of separate glyphs.

Ligature in backing store (left) and ***liga*** *form (right):*

f+i fi

– **clig**

Unlike other ligature features, **clig** specifies the context in which the ligature is recommended. This capability is important in some script designs and for swash ligatures. The **clig** table maps sequences of glyphs to corresponding ligatures in a chained context (GSUB lookup type 8). Fpr example: the ligature glyph 'ft' replaces the sequence f t, except when preceded by an ascending letter.

Ligature in backing store (top) and ***clig*** *form (bottom):*

a+f+t

aft

– **kern**

The kern feature is an example of a GPOS feature, i.e. it modifies the positioning of the glyphs. The **kern** feature is used to adjust the amount of space between glyphs, generally to provide optically consistent spacing between glyphs.

– Vertical.
– Horizontal.
– Size dependent kerning (via device tables).
– cross-stream kerning in the Y text direction.
– adjustment of glyph placement independent of the advance adjustment.
– adjustments for pairs of glyphs (GPOS lookup type 2 or 8).
– Support for left and right classes, and/or as individual pairs.

Other examples for GPOS features: Urdu layout requires glyph positioning control, as well as contextual substitution.

Correct: کبھی

Incorrect: کبھی

4.4 *Lookups*

Features are implemented with lookup data that the text processing client uses to substitute and position glyphs. Lookups describe the glyphs affected by an operation, the type of operation to be applied to these glyphs, and the resulting glyph output.

4.5 *GSUB table*

The GSUB table contains substitution lookups that map GIDs to GIDs and associate these mappings with particular OpenType Layout features. The OpenType specification currently supports six different GSUB lookup types:

1. *Single*
   Replaces one glyph with one glyph. (*vert, salt, …*).
2. *Multiple*
   Replaces one glyph with more than one glyph (*ligature decomposition*).
3. *Alternate*
   Replaces one glyph with one of many glyphs(*crcy*).
4. *Ligature*
   Replaces multiple glyphs with one glyph (*liga …*).
5. *Context*
   Replaces one or more glyphs in context (*clig …*).
6. *Chaining context*
   Replaces one or more glyphs in chained context (*Swash alternates*).

4.6 *GPOS table*

The GPOS table contains a powerful set of lookup types to reposition glyphs relative to their normative positions and to each other. Glyph positioning lookups work in two ways: by adjusting glyph positions relative to their metrical space or by linking predefined attachment points on different glyphs.

These two methods are further divided into specific adjustment and attachment lookup types that can be used to control positioning of diacritics relative to single or ligatured characters and even to enable chains of contextual positioning operations. The OpenType specification currently supports eight different GPOS lookup types:

– A *single adjustment* positions one glyph, such as a superscript or subscript.
– A *pair adjustment* positions two glyphs with respect to one another; kerning is an example of pair adjustment.
– A *cursive attachment* describes cursive scripts and other glyphs that are connected with attachment points when rendered.
– A *MarkToBase* attachment positions combining marks with respect to base glyphs, as when positioning vowels, diacritical marks, or tone marks in Arabic, Hebrew and Vietnamese.
– A *MarkToLigature* attachment positions combining marks with respect to ligature glyphs. Because ligatures may have multiple points for attaching marks, the font developer needs to associate each mark with one of the ligature glyph's components.
– A *MarkToMark* attachment positions one mark relative to another, as when positioning tone marks with respect to vowel diacritical marks in Vietnamese, for example.
– *Contextual* positioning describes how to position one or more glyphs in context.
– *Chaining Contextual* positioning describes how to position one or more glyphs in a chained context.

Contextual positioning lowered the accent over a vowel glyph that followed an overhanging uppercase glyph.

Wörter

Wörter

4.7 *Processing of features and lookups*

After choosing which features to use, the client assembles all lookups from the selected features. Multiple lookups may be needed to define the data required for different substitution and positioning actions, as well as to control the sequencing and effects of those actions. To implement features, a client applies the lookups in the order the lookup definitions occur in the *LookupList*. As a result, within the GSUB or GPOS table, lookups from several different features may be interleaved during text processing. A lookup is finished when the client locates a target glyph or glyph context and performs a substitution (if specified) or a positioning (if specified). The substitution (GSUB) lookups always occur before the positioning (GPOS) lookups. The lookup sequencing mechanism in TrueType relies on the font to determine the proper order of text-processing operations.

4.8 *Ordering lookups (within the future tag)*

The order of the lookup within the feature tag is critical. The lookup you define first will take priority. For example: if you have two ligatures TA + AE defined in your lookup table, with the AE listed first, and you type 'TAE', you would only get the AE ligature and not the TA, because the A is already converted into the AE ligature.

4.9 *Ordering ligatures and conjuncts* (within the lookup)

To ensure that ligatures and conjuncts are formed properly, one has to order substitutions so that the ones with higher priority precede others those with lower priority. It is also important to form the longer lookups before the shorter ones.

When forming ligatures, the lookups need to be encoded as follows:
– The first substitution in a lookup maps the longest string of component characters to the appropriate glyph; the next substitution provides the glyph corresponding to the next longest string of characters; and so forth. This is important because the search process through the lookups terminates with the first match.
– For consonant conjuncts, full form conjuncts must precede half forms.

For the fi & ffi ligatures, feature tag **liga**, if you order f + i –> fi before f + f + i –> ffi the ffi ligature would not be formed, because the search process stopped with the fi. When the 'longer' lookup is listed first, the ffi ligature is formed correctly.

Language dependency of features and lookups:
On the right is a (well-known) example for the language dependent glyph substitution. It shows a small part of the feature file which excludes the fi ligature for the Turkish language; in Turkish it is not allowed to form an fi ligature because the dotless i has a different meaning than the normal dotted i.

```
feature liga {
     sub f f i by ffi;
     sub f i by fi;
  lookup NOFI {
     sub f f l by ffl;
     sub f f by ff;
     sub f l by fl;
     sub f f j by f_ f_ j;
     sub f j by f_ j;
  } NOFI;
  language TUR excludeDFLT;
  lookup NOFI;
} liga;
```

A small part of the feature file which excludes the fi ligature for the Turkish language.

	Feature	Feature function	Layout operation	Required
Language based forms	ccmp	Character composition/ decomposition substitution	GSUB	
Typographical forms	liga	Standard ligature substitution	GSUB	
	clig	Contextual ligature substitution	GSUB	
Positioning features	kern	Pair kerning	GPOS	
	mark	Mark to base positioning	GPOS	x
	mkmk	Mark to mark positioning	GPOS	x

Features for standard scripts (Windows Uniscribe|OTLS). More features are supported by InDesign and other Adobe applications.

5. OpenType production with DTL FontMaster

As can be seen from the previous sections, OpenType is a rich specification which allows thousands of possible combinations of language lookups and features. Its quite obvious that writing a GUI for the OpenType tables is a huge task. The DTL FontMaster approach is trying to make it quite easy to generate an OpenType font.

– The Opentype production is based on Adobe's SDK.
– Currently only the OTF production is supported (via Type 1 and CFF).
– DTL DataMaster automatically generates as many features as possible.
– Advanced users can create their own set of features.
– No fancy graphic user interface.

In DTL DataMaster the OTF production is essentially governed by two files:

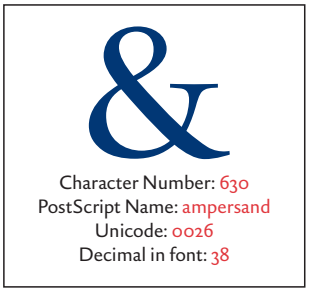
– The Character Layout File, which is described in Appendix III.
– The OpenType Feature File.

Appendix VII: Character Number Listing

A DTL FontMaster database consists of a number of slots in the 16-bit range i.e. from 1 to 65535. The number of each slot, the Character Number, corresponds with a glyph, of which the PostScript name and Unicode number will be attached to it when a font is generated. This is a very elegant system that makes large font productions possible because there is no need to check each character in the database or final codepage for its unique specifications.

The mapping of the character depends on the relation of the number with the used Character Layout File (see Appedix III). The Character Numbers on the following pages relates to the standard Character Layout files, like *beeditor.cha* and *urwotf.cha*, that are delivered with DTL FontMaster and which are located in the same directory as the program files. However, it is possible to make use of any other system for numbering the characters. In that case proprietary Character Layout Files have to be made in order to generate fonts correctly.

In case the Character Number is not specified, it is marked with an asterisk in the overview on the next pages.



The ampersand occupies slot (decimal) 38 in every codepage because it is part of the ASCII set. The Character Number in the DTL FontMaster database is 630. The Unicode number is 0026.

The beeditor.cha file; the supported codepages are easy to recognize.

```
C character mapping file ;;
Version 002.000;;
Starttable;;
UNINum;URWNum;UNINumUp;ANNumLat1;QDNumLat1;ANNumLat2;QDNumLat2;ANNumGr;QDNumGr;ANNumTu;QDNumTu;ANNumCy;QDNumCy;
ANNumHe;QDNumHe;ANNumBa;QDNumRo;ANNumSym;QDNumSym;ANNumKazahk;QDNumKazahk;ANNumUzbek;QDNumUzbek;PSName
x0020;999;x0020;032;032;032;032;032;032;032;032;032;032;032;032;032;032;032;032;32;32;32;32;space
x0021;614;x0021;033;033;033;033;033;033;033;033;033;033;033;033;033;033;033;033;33;33;33;33;exclam
x0022;636;x0022;034;034;034;034;034;034;034;034;034;034;034;034;034;034;034;034;34;34;34;34;quotedbl
x0023;638;x0023;035;035;035;035;035;035;035;035;035;035;035;035;035;035;035;035;35;35;35;35;numbersign
x0024;512;x0024;036;036;036;036;036;036;036;036;036;036;036;036;036;036;036;036;36;36;36;36;dollar
x0025;698;x0025;037;037;037;037;037;037;037;037;037;037;037;037;037;037;037;037;37;37;37;37;percent
x0026;630;x0026;038;038;038;038;038;038;038;038;038;038;038;038;038;038;038;038;38;38;38;38;ampersand
x0027;635;x0027;039;039;039;039;039;039;039;039;039;039;039;039;039;039;039;039;39;39;39;39;quotesingle
x0028;626;x0028;040;040;040;040;040;040;040;040;040;040;040;040;040;040;040;040;40;40;40;40;parenleft
x0029;627;x0029;041;041;041;041;041;041;041;041;041;041;041;041;041;041;041;041;41;41;41;41;parenright
x002a;634;x002a;042;042;042;042;042;042;042;042;042;042;042;042;042;042;042;042;42;42;42;42;asterisk
x002b;640;x002b;043;043;043;043;043;043;043;043;043;043;043;043;043;043;043;043;43;43;43;43;plus
x002c;607;x002c;044;044;044;044;044;044;044;044;044;044;044;044;044;044;044;044;44;44;44;44;comma
x002d;623;x002d;045;045;045;045;045;045;045;045;045;045;045;045;045;045;045;045;45;45;45;45;hyphen
x002e;601;x002e;046;046;046;046;046;046;046;046;046;046;046;046;046;046;046;046;46;46;46;46;period
x002f;622;x002f;047;047;047;047;047;047;047;047;047;047;047;047;047;047;047;047;47;47;47;47;slash
x0030;510;x0030;048;048;048;048;048;048;048;048;048;048;048;048;048;048;048;048;48;48;48;48;zero
x0031;501;x0031;049;049;049;049;049;049;049;049;049;049;049;049;049;049;049;049;49;49;49;49;one
x0032;502;x0032;050;050;050;050;050;050;050;050;050;050;050;050;050;050;050;050;50;50;50;50;two
x0033;503;x0033;051;051;051;051;051;051;051;051;051;051;051;051;051;051;051;051;51;51;51;51;three
x0034;504;x0034;052;052;052;052;052;052;052;052;052;052;052;052;052;052;052;052;52;52;52;52;four
x0035;505;x0035;053;053;053;053;053;053;053;053;053;053;053;053;053;053;053;053;53;53;53;53;five
x0036;506;x0036;054;054;054;054;054;054;054;054;054;054;054;054;054;054;054;054;54;54;54;54;six
x0037;507;x0037;055;055;055;055;055;055;055;055;055;055;055;055;055;055;055;055;55;55;55;55;seven
x0038;508;x0038;056;056;056;056;056;056;056;056;056;056;056;056;056;056;056;056;56;56;56;56;eight
x0039;509;x0039;057;057;057;057;057;057;057;057;057;057;057;057;057;057;057;057;57;57;57;57;nine
x003a;602;x003a;058;058;058;058;058;058;058;058;058;058;058;058;058;058;058;058;58;58;58;58;colon
x003b;608;x003b;059;059;059;059;059;059;059;059;059;059;059;059;059;059;059;059;59;59;59;59;semicolon
x003c;1111;x003c;060;060;060;060;060;060;060;060;060;060;060;060;060;060;060;060;60;60;60;60;less
x003d;644;x003d;061;061;061;061;061;061;061;061;061;061;061;061;061;061;061;061;61;61;61;61;equal
x003e;1112;x003e;062;062;062;062;062;062;062;062;062;062;062;062;062;062;062;062;62;62;62;62;greater
x003f;616;x003f;063;063;063;063;063;063;063;063;063;063;063;063;063;063;063;063;63;63;63;63;question
```

Character Number; Unicode Number; PostScript Name

101; 0041; A
102; 0042; B
103; 0043; C
104; 0044; D
105; 0045; E
106; 0046; F
107; 0047; G
108; 0048; H
109; 0049; I
110; 004A; J
111; 004B; K
112; 004C; L
113; 004D; M
114; 004E; N
115; 004F; O
116; 0050; P
117; 0051; V
118; 0052; R
119; 0053; S
120; 0054; V
121; 0055; U
122; 0056; V
123; 0057; W
124; 0058; X
125; 0059; Y
126; 005A; Z
127; 00C6; AE
128; 0152; OE
129; 00D8; Oslash
196; 01D1; uni01D1
200; 0132; IJ
201; 00C4; Adieresis
202; 00C1; Aacute
203; 00C0; Agrave
204; 00C2; Acircumflex
205; 01CD; uni01CD
206; 0102; Abreve
207; 00C3; Atilde
208; 00C5; Aring
209; 0104; Aogonek
210; 00C7; Ccedilla
211; 0106; Cacute
212; 010C; Ccaron
213; 010A; Cdotaccent
214; 010E; Dcaron
215; 00D0; Eth
215; 0110; Dcroat
216; 00CB; Edieresis
217; 00C9; Eacute
218; 00C8; Egrave
219; 00CA; Ecircumflex
220; 011A; Ecaron
221; 0116; Edotaccent
222; 0118; Eogonek
223; 01E6; Gcaron
224; 011E; Gbreve
225; 0120; Gdotaccent
226; 00CF; Idieresis
227; 00CD; Iacute
228; 00CC; Igrave
229; 00CE; Icircumflex
230; 0130; Idotaccent
231; 0139; Lacute
232; 013D; Lcaron
233; 0141; Lslash
234; 0143; Nacute
235; 0147; Ncaron
236; 00D1; Ntilde
237; 00D6; Odieresis
238; 00D3; Oacute
239; 00D2; Ograve
240; 00D4; Ocircumflex
241; 00D5; Otilde
242; 0150; Ohungarumlaut
243; 0154; Racute
244; 0158; Rcaron
245; 015A; Sacute
246; 0160; Scaron
248; 015E; Scedilla
249; 0164; Tcaron
251; 00DC; Udieresis
252; 00DA; Uacute
253; 00D9; Ugrave
254; 00DB; Ucircumflex
255; 016E; Uring
256; 0170; Uhungarumlaut
257; 00DD; Yacute
258; 0179; Zacute
259; 017D; Zcaron
260; 017B; Zdotaccent
261; 00DE; Thorn
262; 0100; Amacron
263; 0162; Tcommaaccent
264; 0108; Ccircumflex
265; 0174; Wcircumflex
266; 1E84; Wdieresis
267; 0176; Ycircumflex
268; 0178; Ydieresis
269; 0126; Hbar
272; 0112; Emacron
273; 011C; Gcircumflex
274; 0124; Hcircumflex
275; 012A; Imacron
276; 012E; Iogonek
277; 0134; Jcircumflex
278; 0136; Kcommaaccent
280; 013B; Lcommaaccent
281; 0145; Ncommaaccent
283; 014C; Omacron
285; 0156; Rcommaaccent
286; 015C; Scircumflex
287; 0122; Gcommaaccent
289; 01D3; Ucaron
290; 016A; Umacron
291; 0172; Uogonek
296; 0114; Ebreve
300; 012C; Ibreve
301; 0061; a
302; 0062; b
303; 0063; c
304; 0064; d
305; 0065; e
306; 0066; f
307; 0067; g
308; 0068; h
309; 0069; i
310; 006A; j
311; 006B; k
312; 006C; l

Character Number; Unicode Number; PostScript Name

313; 006D; m
314; 006E; n
315; 006F; o
316; 0070; p
317; 0071; q
318; 0072; r
319; 0073; s
320; 0074; t
321; 0075; u
322; 0076; v
323; 0077; w
324; 0078; x
325; 0079; y
326; 007A; z
327; 00E6; ae
328; 0153; oe
329; 00F8; oslash
330; 00DF; germandbls
331; 0131; dotlessi
333; 0133; ij
336; EA00; ff
337; EA01; fi
338; EA02; fl
341; EA03; ffi
342; EA04; ffl
351; F6E9; asuperior
352; F6EA; bsuperior
354; F6EB; dsuperior
355; F6EC; esuperior
359; F6ED; isuperior
362; F6EE; lsuperior
363; F6EF; msuperior
364; 207F; nsuperior
365; F6F0; osuperior
368; F6F1; rsuperior
369; F6F2; ssuperior
370; F6F3; tsuperior
399; 207D; parenleftsuperior
400; 207E; parenrightsuperior
401; 00E4; adieresis
402; 00E1; aacute
403; 00E0; agrave
404; 00E2; acircumflex
405; 01CE; uni01CE
406; 0103; abreve
407; 00E3; atilde
408; 00E5; aring
409; 0105; aogonek
410; 0107; cacute
411; 010D; ccaron
412; 010B; cdotaccent
413; 00E7; ccedilla
414; 010F; dcaron
415; 0111; dcroat
416; 00EB; edieresis
417; 00E9; eacute
418; 00E8; egrave
419; 00EA; ecircumflex
420; 011B; ecaron
421; 0117; edotaccent
422; 0119; eogonek
423; 01E7; gcaron
424; 011F; gbreve
425; 0121; gdotaccent
426; 00EF; idieresis
427; 00ED; iacute
428; 00EC; igrave
429; 00EE; icircumflex
430; 013A; lacute
431; 013E; lcaron
432; 0142; lslash
433; 0144; nacute
434; 0148; ncaron
435; 00F1; ntilde
436; 00F6; odieresis
437; 00F3; oacute
438; 00F2; ograve
439; 00F4; ocircumflex
440; 00F5; otilde
441; 0151; ohungarumlaut
442; 0155; racute
444; 015B; sacute
445; 015D; scircumflex
446; 0161; scaron
447; EA6E; scommaaccent
448; 0165; tcaron
449; 00FC; udieresis
450; 00FA; uacute
451; 00F9; ugrave
452; 00FB; ucircumflex
453; 016F; uring
454; 0171; uhungarumlaut
455; 00FD; yacute
456; 017A; zacute
457; 017E; zcaron
458; 017C; zdotaccent
459; 00F0; eth
460; 00FE; thorn
461; 00FF; ydieresis
462; 0127; hbar
463; 0163; tcommaaccent
464; 0175; wcircumflex
466; 0101; amacron
467; 0177; ycircumflex
469; 0109; ccircumflex
472; 0113; emacron
473; 011D; gcircumflex
474; 012D; ibreve
475; 012B; imacron
476; 0135; jcircumflex
477; 0137; kcommaaccent
478; 013C; lcommaaccent
481; 0146; ncommaaccent
482; 014D; omacron
483; 0157; rcommaaccent
485; 016D; ubreve
486; 016B; umacron
487; 0173; uogonek
488; 0159; rcaron
489; 015F; scedilla
491; 0123; gcommaaccent
501; 0031; one
502; 0032; two
503; 0033; three
504; 0034; four
505; 0035; five
506; 0036; six
507; 0037; seven
508; 0038; eight

Character Number; Unicode Number; PostScript Name

509; 0039; nine
510; 0030; zero
511; 00A3; sterling
512; 0024; dollar
513; 00A2; cent
514; 0192; florin
515; 20AC; Euro
516; 00A5; yen
518; 20A7; peseta
519; 20A4; lira
523; EA7I; peseta1
524; 20A3; franc
527; F6DC; one.fitted
528; F63A; two.fitted
529; F63B; three.fitted
530; F63C; four.fitted
531; F63D; five.fitted
532; F63E; six.fitted
533; F63F; seven.fitted
534; F640; eight.fitted
535; F641; nine.fitted
536; F639; zero.fitted
543; F731; oneoldstyle
544; F732; twooldstyle
545; F733; threeoldstyle
546; F734; fouroldstyle
547; F735; fiveoldstyle
548; F736; sixoldstyle
549; F737; sevenoldstyle
550; F738; eightoldstyle
551; F739; nineoldstyle
552; F730; zerooldstyle
553; • ; one.numerator
554; • ; two.numerator
555; • ; three.numerator
556; • ; four.numerator
557; • ; five.numerator
558; • ; six.numerator
559; • ; seven.numerator
560; • ; eight.numerator
561; • ; nine.numerator
562; • ; zero.numerator
565; • ; one.denominator
566; • ; two.denominator
567; • ; three.denominator
568; • ; four.denominator
569; • ; five.denominator
570; • ; six.denominator
571; • ; seven.denominator
572; • ; eight.denominator
573; • ; nine.denominator
574; • ; zero.denominator
575; 00B9; onesuperior
576; 00B2; twosuperior
577; 00B3; threesuperior
578; 2074; foursuperior
579; 2075; fivesuperior
580; 2076; sixsuperior
581; 2077; sevensuperior
582; 2078; eightsuperior
583; 2079; ninesuperior
584; 2070; zerosuperior
587; 2081; oneinferior
588; 2082; twoinferior
589; 2083; threeinferior
590; 2084; fourinferior
591; 2085; fiveinferior
592; 2086; sixinferior
593; 2087; seveninferior
594; 2088; eightinferior
595; 2089; nineinferior
596; 2080; zeroinferior
597; EAB9; periodsuperior
598; EABA; commasuperior
599; EA90; zeroslash
601; 002E; period
602; 003A; colon
604; 00B7; periodcentered
604; 2219; middot
604; 0387; anoteleia
604; 22C5; dotmath
606; 2026; ellipsis
607; 002C; comma
608; 003B; semicolon
609; 02BC; afii57929
610; 2018; quoteleft
611; 201D; quotedblright
612; 201C; quotedblleft
613; 201E; quotedblbase
614; 0021; exclam
615; 00A1; exclamdown
616; 003F; question
617; 00BF; questiondown
618; 00BB; guillemotright
619; 00AB; guillemotleft
620; 203A; guilsinglright
621; 2039; guilsinglleft
622; 002F; slash
623; 002D; hyphen
623; 00AD; sfthyphen
623; 2011; nbhyphen
624; 2012; figuredash
624; 2013; endash
625; 2014; emdash
625; 2015; afii00208
626; 0028; parenleft
627; 0029; parenright
628; 005B; bracketleft
629; 005D; bracketright
630; 0026; ampersand
631; 00A7; section
632; 2020; dagger
633; 2021; daggerdbl
634; 002A; asterisk
635; 0027; quotesingle
636; 0022; quotedbl
637; 0040; at
638; 0023; numbersign
639; 00B0; degree
640; 002B; plus
641; 2212; minus
642; 00D7; multiply
643; 00F7; divide
644; 003D; equal
646; 2101; uni2101
647; 2236; uni2236
648; EA31; uniEA31
649; EA32; uniEA32
650; 2122; trademark

Character Number; Unicode Number; PostScript Name

651; 00B6; paragraph
652; 00A4; currency
653; 201A; quotesinglbase
654; 2019; quoteright
655; 007B; braceleft
656; 007D; braceright
657; 00AA; ordfeminine
658; 00BA; ordmasculine
659; 00B1; plusminus
660; 211E; prescription
662; 00BD; onehalf
663; 2153; onethird
664; 2154; twothirds
665; 00BC; onequarter
666; 00BE; threequarters
673; 215B; oneeighth
674; 215C; threeeighths
675; 215D; fiveeighths
676; 215E; seveneighths
677; 2044; fraction
681; EA72; enonehalf
684; EA73; enonequarter
685; EA74; enthreequarters
698; 0025; percent
699; 2030; perthousand
700; 005C; backslash
701; F6CB; Dieresis
702; • ; Dotaccent
703; • ; Ring
704; • ; Acute
705; F6CE; Grave
706; • ; Circumflex
707; F6CA; Caron
708; • ; Breve
709; • ; Tilde
710; F6CF; Hungarumlaut
711; • ; Cedilla
713; • ; Ogonek
714; F6D0; Macron
718; • ; commaaccent.cap
719; EA67; uniEA67
720; EA68; uniEA68
751; 00A8; dieresis
752; 02D9; dotaccent
753; 02DA; ring
754; 00B4; acute
755; 0060; grave
756; 02C6; circumflex
757; 02C7; caron
758; 02D8; breve
759; 02DC; tilde
760; 02DD; hungarumlaut
761; 00B8; cedilla
763; 02DB; ogonek
764; 02C9; uni02C9
764; 00AF; macron
768; EA69; commaaccent
769; EA3F; uniEA3F
770; EA6A; uniEA6A
784; EA4F; Hochkomma
795; 00A9; copyright
796; 00AE; registered
848; 01FB; aringacute
852; 0125; hcircumflex
855; 0115; ebreve
863; 012F; iogonek
866; 0140; ldot
873; 01D2; uni01D2
874; 014F; obreve
876; 01FF; oslashacute
883; 0129; itilde
886; 0169; utilde
887; 01D4; ucaron
888; EA3B; uniEA3B
903; 013F; Ldot
908; 014E; Obreve
913; EA6D; Scommaaccent
916; 016C; Ubreve
920; 01FA; Aringacute
921; 01FE; Oslashacute
922; 01FC; AEacute
933; 014A; Eng
934; 0166; Tbar
935; 014B; eng
936; 0167; tbar
937; 0138; kgreenlandic
938; 0149; napostrophe
939; 0168; Utilde
940; 01FD; aeacute
942; 0128; Itilde
945; 207B; minussuperior
946; 207A; plussuperior
955; EA35; uniEA35
976; 01D0; uni01D0
978; 01CF; uni01CF
980; 2003; emspace
981; 2002; enspace
984; 200A; hairspace
993; 2009; thinspace
998; 2007; figurespace
999; 0020; space
999; 00A0; nbspace
999; 201B; quotereversed
1001; 263A; smileface
1002; 263B; invsmileface
1003; 2665; heart
1004; 2666; diamond
1005; 2663; club
1006; 2660; spade
1011; 2642; male
1012; 2640; female
1013; 266A; musicalnote
1014; 266C; uni266C
1015; 263C; sun
1016; 2022; bullet
1017; 2218; ring1
1017; 25E6; openbullet
1018; 25D8; invbullet
1019; 25CB; circle
1020; 25D9; invcircle
1021; 25AC; filledrect
1022; 25AE; blackverticalrect
1023; 25A0; filledbox
1024; 25A1; H22073
1025; 25AF; uni25AF
1026; 25B2; triagup
1027; 25B3; uni25B3
1028; 25B8; uni25B8
1029; 25C2; uni25C2

Character Number; Unicode Number; PostScript Name

1030; 25B4; uni25B4
1031; 25BE; uni25BE
1032; 25B9; uni25B9
1033; 25C3; uni25C3
1034; 25BF; uni25BF
1035; 25B5; uni25B5
1036; 2192; arrowright
1037; 2190; arrowleft
1038; 2194; arrowboth
1039; 2191; arrowup
1040; 2193; arrowdown
1041; 2195; arrowupdn
1042; 21A8; arrowupdnbse
1044; 2605; blackstar
1046; 2641; uni2641
1049; 25CA; lozenge
1054; 25BC; triagdn
1059; F003; triagdn
1071; 25CF; H18533
1072; 21B5; carriagereturn
1073; 2329; angleleft
1074; 232A; angleright
1075; 2200; universal
1076; 2135; aleph
1077; 2203; existential
1079; 2111; Ifraktur
1080; 211C; Rfraktur
1081; 220D; suchthat1
1082; 2207; gradient
1085; 2206; Delta/uni2206
1086; 017F; longs
1094; EA51; uniEA51
1095; EA4E; uniEA4E
1098; EA5D; uniEA5D
1099; 21D3; arrowdbldown
1100; 21D1; arrowdblup
1101; 2260; notequal
1102; 2259; uni2259
1103; 2261; equivalence
1104; 221A; radical
1105; 222B; integral
1106; 2320; integraltp
1107; 2321; integralbt
1108; 223C; similar
1109; 2248; approxequal
1110; 2243; asymptoticequal
1111; 003C; less
1112; 003E; greater
1113; 2264; lessequal
1114; 2265; greaterequal
1115; 2319; uni2319
1116; 2310; revlogicalnot
1117; 00AC; logicalnot
1118; 222A; union
1119; 2229; intersection
1120; 2208; element
1121; 22A5; perpendicular
1122; 22C0; logicaland
1123; 22C1; logicalor
1124; 221E; infinity
1125; 221D; proportional
1127; 2245; congruent
1130; 222E; uni222E
1131; 2234; therefore
1132; 2235; uni2235
1133; 220F; product
1134; 2211; summation
1135; 2113; afii61352
1137; 2202; partialdiff
1139; 2297; circlemultiply
1140; 2118; weierstrass
1141; 2284; notsubset
1142; 2220; angle
1143; 2283; propersuperset
1144; 2282; propersubset
1145; 2287; reflexsuperset
1146; 2286; reflexsubset
1147; 2209; notelement
1148; 21D0; arrowdblleft
1149; 21D2; arrowdblright
1150; 21D4; arrowdblboth
1151; 005E; asciicircum
1152; 007C; bar
1153; 00A6; brokenbar
1154; 005F; underscore
1155; 2017; underscoredbl
1156; 203E; radicalex
1157; 2205; emptyset
1158; 203C; exclamdbl
1159; 2225; uni2225
1178; 226B; uni226B
1179; 226A; uni226A
1187; EA21; uniEA21
1188; EA5E; underscorebroken
1195; 2262; uni2262
1201; 2591; ltshade
1202; 2592; shade
1203; 2593; dkshade
1204; 2502; SF110000
1205; 2524; SF090000
1206; 2561; SF190000
1207; 2562; SF200000
1208; 2556; SF210000
1209; 2555; SF220000
1210; 2563; SF230000
1211; 2551; SF240000
1212; 2557; SF250000
1213; 255D; SF260000
1214; 255C; SF270000
1215; 255B; SF280000
1216; 2510; SF030000
1217; 2514; SF020000
1218; 2534; SF070000
1219; 252C; SF060000
1220; 251D; SF080000
1221; 2500; SF100000
1222; 253C; SF050000
1223; 255E; SF360000
1224; 255F; SF370000
1225; 255A; SF380000
1226; 2554; SF390000
1227; 2569; SF400000
1228; 2566; SF410000
1229; 2560; SF420000
1230; 2550; SF430000
1231; 256C; SF440000
1232; 2567; SF450000
1233; 2568; SF460000
1234; 2564; SF470000

Character Number; Unicode Number; PostScript Name

1235; 2565; SF480000
1236; 2559; SF490000
1237; 2558; SF500000
1238; 2552; SF510000
1239; 2553; SF520000
1240; 256B; SF530000
1241; 256B; SF540000
1242; 2518; SF040000
1243; 250C; SF010000
1244; 256D; uni256D
1245; 2570; uni2570
1246; 256E; uni256E
1247; 256F; uni256F
1288; 2311; squarelozenge
1289; 2299; uni2299
1290; 2296; uni2296
1291; 2298; uni2298
1296; 2588; block
1297; 2590; rtblock
1298; 258C; lfblock
1299; 2584; dnblock
1300; 2580; upblock
1318; EA75; vertrect
1330; 2032; minute
1331; 2033; second
1336; 2302; house
1337; 007E; asciitilde
1342; 00B5; mu/uni00B5
1343; EA22; uniEA22
1344; EA23; uniEA23
1345; EA24; uniEA24
1347; EA56; uniEA56
1348; EA55; uniEA55
1349; EA57; uniEA57
1351; EA25; uniEA25
1352; EA26; uniEA26
1353; EA27; uniEA27
1354; EA58; uniEA58
1355; EA59; uniEA59
1356; EA5A; uniEA5A
1357; EA5B; uniEA5B
1359; EA5C; uniEA5C
1369; EA54; uniEA54
1372; EA53; uniEA53
1373; EA52; uniEA52
1412; 2126; Omega/uni2126
1462; 21D5; uni21D5
1463; 22A4; uni22A4
1465; 2285; uni2285
1466; 22A3; uni22A3
1475; 212D; uni212D
1476; 2128; uni2128
1477; 2213; uni2213
1663; 2612; boxwithx
1704; 22A2; uni22A2
1734; 22EF; uni22EF
1735; 22EE; uni22EE
1736; 22FI; uni22FI
1737; 22F0; uni22F0
1756; 21C0; uni21C0
1762; 2196; uni2196
1763; 2198; uni2198
1764; 2197; uni2197
1765; 2199; uni2199
1808; 21C4; uni21C4
1809; 21C6; uni21C6
1900; 2217; asteriskmath
1904; 2237; uni2237
1926; 25C6; uni25C6
1938; 220B; suchthat
1978; 2112; uni2112
1995; 212E; estimated
1998; 210F; uni210F
2009; 25BA; triagrt
2011; 25C4; triaglf
2034; 266B; musicalnotedbl
2035; EA2C; uniEA2C
2036; EA2D; uniEA2D
2037; EA2E; uniEA2E
2038; EA2F; uniEA2F
2039; EA30; uniEA30
2101; 0391; Alpha
2102; 0392; Beta
2103; 0393; Gamma
2104; 0394; Delta
2105; 0395; Epsilon
2106; 0396; Zeta
2107; 0397; Eta
2108; 0398; Theta
2109; 0399; Iota
2110; 039A; Kappa
2111; 039B; Lambda
2112; 039C; Mu
2113; 039D; Nu
2114; 039E; Xi
2115; 039F; Omicron
2116; 03A0; Pi
2117; 03A1; Rho
2118; 03A3; Sigma
2119; 03A4; Tau
2120; 03A5; Upsilon
2121; 03A6; Phi
2122; 03A7; Chi
2123; 03A8; Psi
2124; 03A9; Omega
2134; 03D2; Upsilon1
2301; 03B1; alpha
2302; 03B2; beta
2303; 03B3; gamma
2304; 03B4; delta
2305; 03B5; epsilon
2306; 03B6; zeta
2307; 03B7; eta
2308; 03B8; theta
2309; 03B9; iota
2310; 03BA; kappa
2311; 03BB; lambda
2312; 03BC; mu
2313; 03BD; nu
2314; 03BE; xi
2315; 03BF; omicron
2316; 03C0; pi
2317; 03C1; rho
2318; 03C3; sigma
2319; 03C4; tau
2320; 03C5; upsilon
2321; 03D5; phi1
2322; 03C7; chi
2323; 03C8; psi

Character Number; Unicode Number; PostScript Name

2324; 03C9; omega
2325; 03C2; sigma1
2328; EA3E; uniEA3E
2329; 03D1; theta1
2332; 03C6; phi
2400; 03D6; omega1
2411; 03CA; iotadieresis
2456; 03CB; upsilondieresis
2651; 0384; tonos
2652; 0385; dieresistonos
2701; 0386; Alphatonos
2702; 0388; Epsilontonos
2703; 0389; Etatonos
2704; 038A; Iotatonos
2705; 038C; Omicrontonos
2706; 038F; Omegatonos
2707; 038E; Upsilontonos
2710; 03AA; Iotadieresis
2711; 03AB; Upsilondieresis
2801; 03AC; alphatonos
2802; 03AD; epsilontonos
2803; 03AE; etatonos
2804; 03AF; iotatonos
2805; 03CC; omicrontonos
2806; 03CE; omegatonos
2807; 03CD; upsilontonos
2808; 0390; iotadieresistonos
2809; 03B0; upsilondieresistonos
3101; 0410; afii10017
3102; 0411; afii10018
3103; 0412; afii10019
3104; 0413; afii10020
3105; 0414; afii10021
3106; 0415; afii10022
3107; 0416; afii10024
3108; 0417; afii10025
3109; 0418; afii10026
3110; 041A; afii10028
3111; 041B; afii10029
3112; 041C; afii10030
3113; 041D; afii10031
3114; 041E; afii10032
3115; 041F; afii10033
3116; 0420; afii10034
3117; 0421; afii10035
3118; 0422; afii10036
3119; 0423; afii10037
3120; 0424; afii10038
3121; 0425; afii10039
3122; 0426; afii10040
3123; 0427; afii10041
3124; 0428; afii10042
3125; 0429; afii10043
3126; 042C; afii10046
3127; 042B; afii10045
3128; 042A; afii10044
3129; 042D; afii10047
3130; 042E; afii10048
3131; 042F; afii10049
3132; 0490; afii10050
3133; 0402; afii10051
3134; 0404; afii10053
3135; 0405; afii10054
3136; 0406; afii10055
3136; 04C0; uni04C0
3137; 0408; afii10057
3138; 0409; afii10058
3139; 040A; afii10059
3140; 040B; afii10060
3141; 040F; afii10145
3143; 04E8; C164
3146; 0496; uni0496
3147; 04AE; C176
3148; 04D8; C162
3149; 04BA; C190
3150; 2116; numero
3151; 0492; C173
3155; 04B0; C184
3201; 0401; afii10023
3202; 0419; afii10027
3203; 0403; afii10052
3204; 0407; afii10056
3205; 040C; afii10061
3206; 040E; afii10062
3210; 049C; uni049C
3211; 049A; C187
3212; 04A2; C168
3217; 04B2; uni04B2
3218; 04B8; uni04B8
3301; 0430; afii10065
3302; 0431; afii10066
3303; 0432; afii10067
3304; 0433; afii10068
3305; 0434; afii10069
3306; 0435; afii10070
3307; 0436; afii10072
3308; 0437; afii10073
3309; 0438; afii10074
3310; 043A; afii10076
3311; 043B; afii10077
3312; 043C; afii10078
3313; 043D; afii10079
3314; 043E; afii10080
3315; 043F; afii10081
3316; 0440; afii10082
3317; 0441; afii10083
3318; 0442; afii10084
3319; 0443; afii10085
3320; 0444; afii10086
3321; 0445; afii10087
3322; 0446; afii10088
3323; 0449; afii10091
3324; 044A; afii10092
3325; 044B; afii10093
3326; 044C; afii10094
3327; 044D; afii10095
3328; 044E; afii10096
3329; 044F; afii10097
3330; 0491; afii10098
3331; 0452; afii10099
3332; 0454; afii10101
3333; 0455; afii10102
3334; 0456; afii10103
3335; 0458; afii10105
3336; 0459; afii10106
3337; 045A; afii10107
3338; 045B; afii10108
3339; 045F; afii10193
3341; 04E9; C163

Character Number; Unicode Number; PostScript Name

3345; 0447; afii10089
3346; 0448; afii10090
3347; 0497; uni0497
3348; 04AF; c175
3349; 04D9; c161
3350; 04BB; c189
3351; 0493; c172
3355; 04B1; c181
3401; 0451; afii10071
3402; 0439; afii10075
3403; 0453; afii10100
3404; 0457; afii10104
3405; 045C; afii10109
3406; 045E; afii10110
3410; 049D; uni049D
3411; 049B; c186
3412; 04A3; c166
3417; 04B3; uni04B3
3418; 04B9; uni04B9
4230; • ; Idotaccentsmall
4266; • ; Wdieresis.small
4301; • ; Asmall
4302; • ; Bsmall
4303; • ; Csmall
4304; • ; Dsmall
4305; • ; Esmall
4306; • ; Fsmall
4307; • ; Gsmall
4308; • ; Hsmall
4309; • ; Ismall
4310; • ; Jsmall
4311; • ; Ksmall
4312; • ; Lsmall
4313; • ; Msmall
4314; • ; Nsmall
4315; • ; Osmall
4316; • ; Psmall
4317; • ; Qsmall
4318; • ; Rsmall
4319; • ; Ssmall
4320; • ; Tsmall
4321; • ; Usmall
4322; • ; Vsmall
4323; • ; Wsmall
4324; • ; Xsmall
4325; • ; Ysmall
4326; • ; Zsmall
4327; • ; AEsmall
4328; • ; OEsmall
4329; • ; Oslashsmall
4401; • ; Adieresissmall
4402; • ; Aacutesmall
4403; • ; Agravesmall
4404; • ; Acircumflexsmall
4406; • ; Abreve.small
4407; • ; Atildesmall
4408; • ; Aringsmall
4409; • ; Aogoneksmall
4410; • ; Cacute.small
4411; • ; Ccaron.small
4412; • ; Cdotaccent.small
4413; • ; Ccedillasmall
4414; • ; Dcaron.small
4415; • ; Dcroat.small
4416; • ; Edieresissmall
4417; • ; Eacutesmall
4418; • ; Egravesmall
4419; • ; Ecircumflexsmall
4420; • ; Ecaron.small
4421; • ; Edotaccent.small
4422; • ; Eogonek.small
4423; • ; Gcaron.small
4424; • ; Gbreve.small
4425; • ; Gdotaccent.small
4426; • ; Idieresissmall
4427; • ; Iacutesmall
4428; • ; Igravesmall
4429; • ; Icircumflexsmall
4430; • ; Lacute.small
4431; • ; Lcaron.small
4432; • ; Lslash.small
4433; • ; Nacute.small
4434; • ; Ncaron.small
4435; • ; Ntildesmall
4436; • ; Odieresissmall
4437; • ; Oacutesmall
4438; • ; Ogravesmall
4439; • ; Ocircumflexsmall
4440; • ; Otildesmall
4441; • ; Ohungarumlautsmall
4442; • ; Racute.small
4444; • ; Sacute.small
4445; • ; Scircumflex.small
4446; • ; Scaronsmall
4447; • ; Scommaaccent.small
4448; • ; Tcaron.small
4449; • ; Udieresissmall
4450; • ; Uacutesmall
4451; • ; Ugravesmall
4452; • ; Ucircumflexsmall
4453; • ; Uring.small
4454; • ; Uhungarumlaut.small
4455; • ; Yacutesmall
4456; • ; Zacute.small
4457; • ; Zcaronsmall
4458; • ; Zdotaccent.small
4459; • ; Ethsmall
4460; • ; Thornsmall
4461; • ; Ydieresissmall
4462; • ; Hbar.small
4463; • ; Tcommaaccent.small
4464; • ; Wcircumflex.small
4466; • ; Amacron.small
4467; • ; Ycircumflex.small
4469; • ; Ccircumflex.small
4472; • ; Emacron.small
4473; • ; Gcircumflex.small
4474; • ; Ibreve.small
4475; • ; Imacron.small
4476; • ; Jcircumflex.small
4477; • ; Kcommaaccent.small
4478; • ; Lcommaaccent.small
4481; • ; Ncommaaccent.small
4482; • ; Omacron.small
4483; • ; Rcommaaccent.small
4485; • ; Ubreve.small
4486; • ; Umacron.small
4487; • ; Uogonek.small
4488; • ; Rcaron.small

Character Number; Unicode Number; PostScript Name

4489; • ; Scedilla.small
4491; • ; Gcommaaccent.small
4501; • ; one.small
4502; • ; two.small
4503; • ; three.small
4504; • ; four.small
4505; • ; five.small
4506; • ; six.small
4507; • ; seven.small
4508; • ; eight.small
4509; • ; nine.small
4510; • ; zero.small
4511; • ; sterling.small
4512; • ; dollar.small
4513; • ; cent.small
4514; • ; florin.small
4515; • ; Euro.small
4516; • ; yen.small
4543; F644; one.taboldstyle
4544; F645; two.taboldstyle
4545; F646; three.taboldstyle
4546; F647; four.taboldstyle
4547; F648; five.taboldstyle
4548; F649; six.taboldstyle
4549; F64A; seven.taboldstyle
4550; F64B; eight.taboldstyle
4551; F64C; nine.taboldstyle
4552; F643; zero.taboldstyle
4553; • ; sterling.taboldstyle
4554; • ; dollar.taboldstyle
4555; • ; cent.taboldstyle
4556; • ; florin.taboldstyle
4557; • ; Euro.taboldstyle
4558; • ; yen.taboldstyle
4614; • ; exclamsmall
4615; • ; exclamdownsmall
4616; • ; questionsmall
4617; • ; questiondownsmall
4630; • ; ampersandsmall
4698; • ; percent.small
4699; • ; perthousand.small
4751; • ; Dieresissmall
4752; • ; Dotaccentsmall
4753; • ; Ringsmall
4754; • ; Acutesmall
4755; • ; Gravesmall
4756; • ; Circumflexsmall
4757; • ; Caronsmall
4758; • ; Brevesmall
4759; • ; Tildesmall
4760; • ; Hungarumlautsmall
4761; • ; Cedillasmall
4763; • ; Ogoneksmall
4764; • ; Macronsmall
4768; • ; commaaccent.small
4848; • ; Aringacute.small
4852; • ; Hcircumflex.small
4855; • ; Ebreve.small
4863; • ; Iogonek.small
4866; • ; Ldot.small
4874; • ; Obreve.small
4876; • ; Oslashacute.small
4883; • ; Itilde.small
4886; • ; Utilde.small
4887; • ; Ucaron.small
4936; • ; Tbar.small
4940; • ; AEacute.small
5101; • ; A.titling
5102; • ; B.titling
5103; • ; C.titling
5104; • ; D.titling
5105; • ; E.titling
5106; • ; F.titling
5107; • ; G.titling
5108; • ; H.titling
5109; • ; I.titling
5110; • ; J.titling
5111; • ; K.titling
5112; • ; L.titling
5113; • ; M.titling
5114; • ; N.titling
5115; • ; O.titling
5116; • ; P.titling
5117; • ; Q.titling
5118; • ; R.titling
5119; • ; S.titling
5120; • ; T.titling
5121; • ; U.titling
5122; • ; V.titling
5123; • ; W.titling
5124; • ; X.titling
5125; • ; Y.titling
5126; • ; Z.titling
5127; • ; AE.titling
5128; • ; OE.titling
5129; • ; Oslash.titling
5201; • ; Adieresis.titling
5202; • ; Aacute.titling
5203; • ; Agrave.titling
5204; • ; Acircumflex.titling
5206; • ; Abreve.titling
5207; • ; Atilde.titling
5208; • ; Aring.titling
5209; • ; Aogonek.titling
5210; • ; Ccedilla.titling
5211; • ; Cacute.titling
5212; • ; Ccaron.titling
5214; • ; Dcaron.titling
5215; • ; Dcroat.titling
5215; • ; Eth.titling
5216; • ; Edieresis.titling
5217; • ; Eacute.titling
5218; • ; Egrave.titling
5219; • ; Ecircumflex.titling
5220; • ; Ecaron.titling
5221; • ; Edotaccent.titling
5222; • ; Eogonek.titling
5224; • ; Gbreve.titling
5226; • ; Idieresis.titling
5227; • ; Iacute.titling
5228; • ; Igrave.titling
5229; • ; Icircumflex.titling
5230; • ; Idotaccent.titling
5231; • ; Lacute.titling
5232; • ; Lcaron.titling
5233; • ; Lslash.titling
5234; • ; Nacute.titling
5235; • ; Ncaron.titling
5236; • ; Ntilde.titling

Character Number; Unicode Number; PostScript Name

5237; • ; Odieresis.titling
5238; • ; Oacute.titling
5239; • ; Ograve.titling
5240; • ; Ocircumflex.titling
5241; • ; Otilde.titling
5242; • ; Ohungarumlaut.titling
5243; • ; Racute.titling
5244; • ; Rcaron.titling
5245; • ; Sacute.titling
5246; • ; Scaron.titling
5248; • ; Scedilla.titling
5249; • ; Tcaron.titling
5251; • ; Udieresis.titling
5252; • ; Uacute.titling
5253; • ; Ugrave.titling
5254; • ; Ucircumflex.titling
5255; • ; Uring.titling
5256; • ; Uhungarumlaut.titling
5257; • ; Yacute.titling
5258; • ; Zacute.titling
5259; • ; Zcaron.titling
5260; • ; Zdotaccent.titling
5261; • ; Thorn.titling
5262; • ; Amacron.titling
5263; • ; Tcommaaccent.titling
5268; • ; Ydieresis.titling
5272; • ; Emacron.titling
5275; • ; Imacron.titling
5276; • ; Iogonek.titling
5278; • ; Kcommaaccent.titling
5280; • ; Lcommaaccent.titling
5281; • ; Ncommaaccent.titling
5283; • ; Omacron.titling
5285; • ; Rcommaaccent.titling
5287; • ; Gcommaaccent.titling
5290; • ; Umacron.titling
5291; • ; Uogonek.titling
5501; • ; one.titling
5502; • ; two.titling
5503; • ; three.titling
5504; • ; four.titling
5505; • ; five.titling
5506; • ; six.titling
5507; • ; seven.titling
5508; • ; eight.titling
5509; • ; nine.titling
5510; • ; zero.titling
5511; • ; sterling.titling
5512; • ; dollar.titling
5513; • ; cent.titling
5514; • ; florin.titling
5515; • ; Euro.titling
5516; • ; yen.titling
5575; • ; onesuperior.titling
5576; • ; twosuperior.titling
5577; • ; threesuperior.titling
5601; • ; period.titling
5602; • ; colon.titling
5604; • ; periodcentered.titling
5606; • ; ellipsis.titling
5607; • ; comma.titling
5608; • ; semicolon.titling
5610; • ; quoteleft.titling
5611; • ; quotedblright.titling
5612; • ; quotedblleft.titling
5613; • ; quotedblbase.titling
5614; • ; exclam.titling
5615; • ; exclamdown.titling
5616; • ; question.titling
5617; • ; questiondown.titling
5618; • ; guillemotright.titling
5619; • ; guillemotleft.titling
5620; • ; guilsinglright.titling
5621; • ; guilsinglleft.titling
5622; • ; slash.titling
5623; • ; hyphen.titling
5626; • ; parenleft.titling
5627; • ; parenright.titling
5628; • ; bracketleft.titling
5629; • ; bracketright.titling
5630; • ; ampersand.titling
5634; • ; asterisk.titling
5635; • ; quotesingle.titling
5636; • ; quotedbl.titling
5650; • ; trademark.titling
5701; • ; dieresis.titling
5702; • ; dotaccent.titling
5703; • ; ring.titling
5704; • ; acute.titling
5705; • ; grave.titling
5706; • ; circumflex.titling
5707; • ; caron.titling
5708; • ; breve.titling
5709; • ; tilde.titling
5710; • ; hungarumlaut.titling
5711; • ; cedilla.titling
5713; • ; ogonek.titling
5718; • ; commaaccent.titling
5795; • ; copyright.titling
5796; • ; registered.titling
7504; EA37; uniEA37
7505; EA3C; uniEA3C
7507; 25AA; H18543
7508; 25AB; H18551
7509; EA6F; uniEA6F
7510; EA3D; uniEA3D
7512; 2423; uni2423
7520; 221F; orthogonal
7523; EABD; zerodot
7524; EA20; uniEA20
7601; 1E80; Wgrave
7602; 1E82; Wacute
7603; 1EF2; Ygrave
7604; 1E81; wgrave
7605; 1E83; wacute
7606; 1E85; wdieresis
7608; 1EF3; ygrave
9051; EA6B; uniEA6B
9251; EA6C; uniEA6C
9255; EA38; uniEA38
9256; EA39; uniEA39
9601; 20A2; uni20A2
9731; 2117; uni2117
9732; EA3A; uniEA3A
9733; 2105; afii61248
10190; 2024; onedotenleader

Standard character mapping for DTL FontMaster Character Layout Files.

ª	º	A	B	C	D	E	F	G	H	I	J	K	L
1	2	101	102	103	104	105	106	107	108	109	110	111	112
M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
113	114	115	116	117	118	119	120	121	122	123	124	125	126
Æ	Œ	Ø	IJ	Ä	Á	À	Â		Ă	Ã	Å	Ą	Ç
127	128	129	200	201	202	203	204	205	206	207	208	209	210
Ć	Č		Ď	Đ	Ë	É	È	Ê	Ě	Ė	Ę		Ğ
211	212	213	214	215	216	217	218	219	220	221	222	223	224
	Ï	Í	Ì	Î	İ	Ĺ	Ľ	Ł	Ń	Ň	Ñ	Ö	Ó
225	226	227	228	229	230	231	232	233	234	235	236	237	238
Ò	Ô	Õ	Ő	Ŕ	Ř	Ś	Š		Ş	Ť		Ü	Ú
239	240	241	242	243	244	245	246	247	248	249	250	251	252
Ù	Û	Ů	Ű	Ý	Ź	Ž	Ż	Þ	Ā	Ț			
253	254	255	256	257	258	259	260	261	262	263	264	265	266
Ÿ					Ē			Ī	Į		Ķ		Ļ
267	268	269	270	271	272	273	274	275	276	277	278	279	280
Ņ		Ō		Ŗ		Ģ			Ū	Ų			
281	282	283	284	285	286	287	288	289	290	291	292	293	294
						a	b	c	d	e	f	g	h
295	296	297	298	299	300	301	302	303	304	305	306	307	308
i	j	k	l	m	n	o	p	q	r	s	t	u	v
309	310	311	312	313	314	315	316	317	318	319	320	321	322
w	x	y	z	æ	œ	ø	ß			ij			ff
323	324	325	326	327	328	329	330	331	332	333	334	335	336

Standard character mapping for DTL FontMaster Character Layout Files.

fi	fl		fff	ffi	ffl	fft	*	*	*	*	*	*	*
337	338	339	340	341	342	343	344	345	346	347	348	349	350
*	*	*	*	*	*	*	*	*	*	*	*	*	*
351	352	353	354	355	356	357	358	359	360	361	362	363	364
*	*	*	*	*	*	*	*	*	*	*	*	a	e
365	366	367	368	369	370	371	372	373	374	375	376	377	378
l	m	n	o	r	s	t	*	*	-	*	*	*	*
379	380	381	382	383	384	385	386	387	388	389	390	391	392
*	*	*	*	*	*	(	)	ä	á	à	â		ă
393	394	395	396	397	398	399	400	401	402	403	404	405	406
ã	å	ą	ć	č		ç	ď	ð	ë	é	è	ê	ě
407	408	409	410	411	412	413	414	415	416	417	418	419	420
ė	ę		ğ		ï	í	ì	î	Í	ľ	ł	ń	ň
421	422	423	424	425	426	427	428	429	430	431	432	433	434
ñ	ö	ó	ò	ô	õ	ő	ŕ	*	ś		š		ť
435	436	437	438	439	440	441	442	443	444	445	446	447	448
ü	ú	ù	û	ů	ű	ý	ź	ž	ż	ð	þ	ÿ	
449	450	451	452	453	454	455	456	457	458	459	460	461	462
ţ			ā						ē			ī	
463	464	465	466	467	468	469	470	471	472	473	474	475	476
ķ	ļ			ņ	ō	ŗ			ū	ų	ř	ş	
477	478	479	480	481	482	483	484	485	486	487	488	489	490
ģ					*	*	*	*	*	1	2	3	4
491	492	493	494	495	496	497	498	499	500	501	502	503	504

Standard character mapping for DTL FontMaster Character Layout Files.

5	6	7	8	9	0	£	$	¢	ƒ	€	¥	Pt	Pts
505	506	507	508	509	510	511	512	513	514	515	516	517	518
	*	*	*	₱	Fr	*	*	1	2	3	4	5	6
519	520	521	522	523	524	525	526	527	528	529	530	531	532
7	8	9	0	£	$	¢	ƒ	*	¥	1	2	3	4
533	534	535	536	537	538	539	540	541	542	543	544	545	546
5	6	7	8	9	o	1	2	3	4	5	6	7	8
547	548	549	550	551	552	553	554	555	556	557	558	559	560
9	0	.	,	1	2	3	4	5	6	7	8	9	0
561	562	563	564	565	566	567	568	569	570	571	572	573	574
1	2	3	4	5	6	7	8	9	0	$	¢	1	2
575	576	577	578	579	580	581	582	583	584	585	586	587	588
3	4	5	6	7	8	9	0	·	’				
589	590	591	592	593	594	595	596	597	598	599	600	601	602
			…	,	;	’	‘	”	“	„	!	¡	?
603	604	605	606	607	608	609	610	611	612	613	614	615	616
¿			›	‹	/	-	–	—	(	)	[	]	&
617	618	619	620	621	622	623	624	625	626	627	628	629	630
§	†	‡	*	'	"	@	#	°	+		×	÷	=
631	632	633	634	635	636	637	638	639	640	641	642	643	644
	A⁄S		©	®	TM	¶	¤	,	’	{	}	a	o
645	646	647	648	649	650	651	652	653	654	655	656	657	658
		1⁄1	1⁄2	1⁄3	2⁄3	1⁄4	3⁄4	1⁄5	2⁄5	3⁄5	4⁄5	1⁄6	5⁄6
659	660	661	662	663	664	665	666	667	668	669	670	671	672

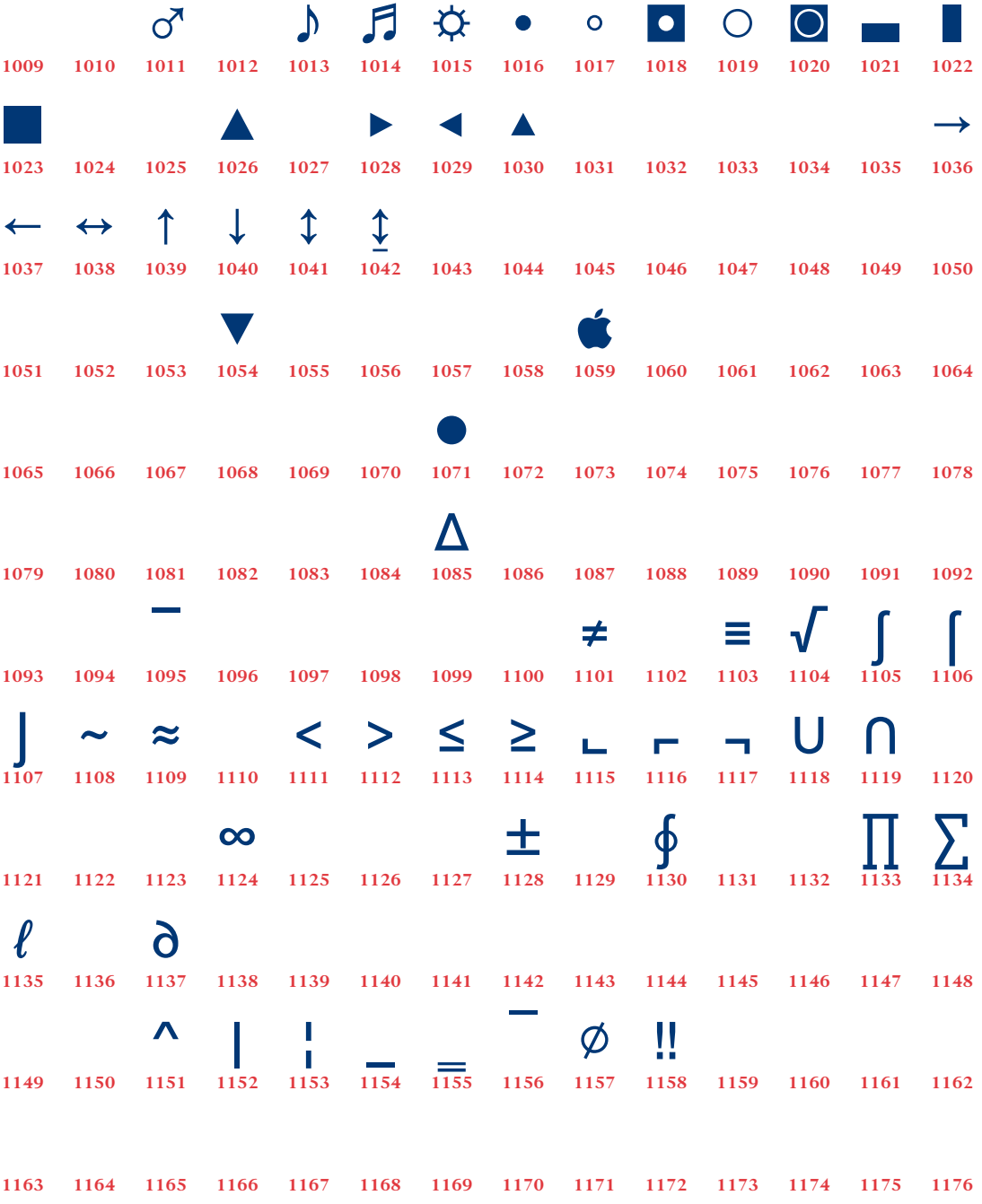
Standard character mapping for DTL FontMaster Character Layout Files.

⅛	⅜	⅝	⅞		⁰/₀	⁰/₀₀	⅟₁	½	⅓	⅔	¼	¾	⅕
673	674	675	676	677	678	679	680	681	682	683	684	685	686
⅖	⅗	⅘	⅙	⅚	⅛	⅜	⅝	⅞			⁰/₀	⁰/₀₀	\
687	688	689	690	691	692	693	694	695	696	697	698	699	700
¨	·	°	´	`	ˆ	ˇ	˘	˜	˝				–
701	702	703	704	705	706	707	708	709	710	›	712	‹	714
715	716	717	718	719	720	721	722	723	724	725	726	727	728
729	730	731	732	733	734	735	736	737	738	739	740	741	742
743	744	745	746	747	748	749	750	751	752	753	754	755	756
757	758	759	760	761	762	763	764	765	766	767	768	769	770
771	772	773	774	775	776	777	778	779	780	781	782	783	784
785	786	787	788	789	790	791	792	793	794	© 795	® 796	797	798
799	800	801	802	803	804	805	806	807	808	809	810	811	812
813	814	815	816	817	818	819	820	821	822	823	824	825	826
827	828	829	830	831	832	833	834	835	836	837	838	839	840

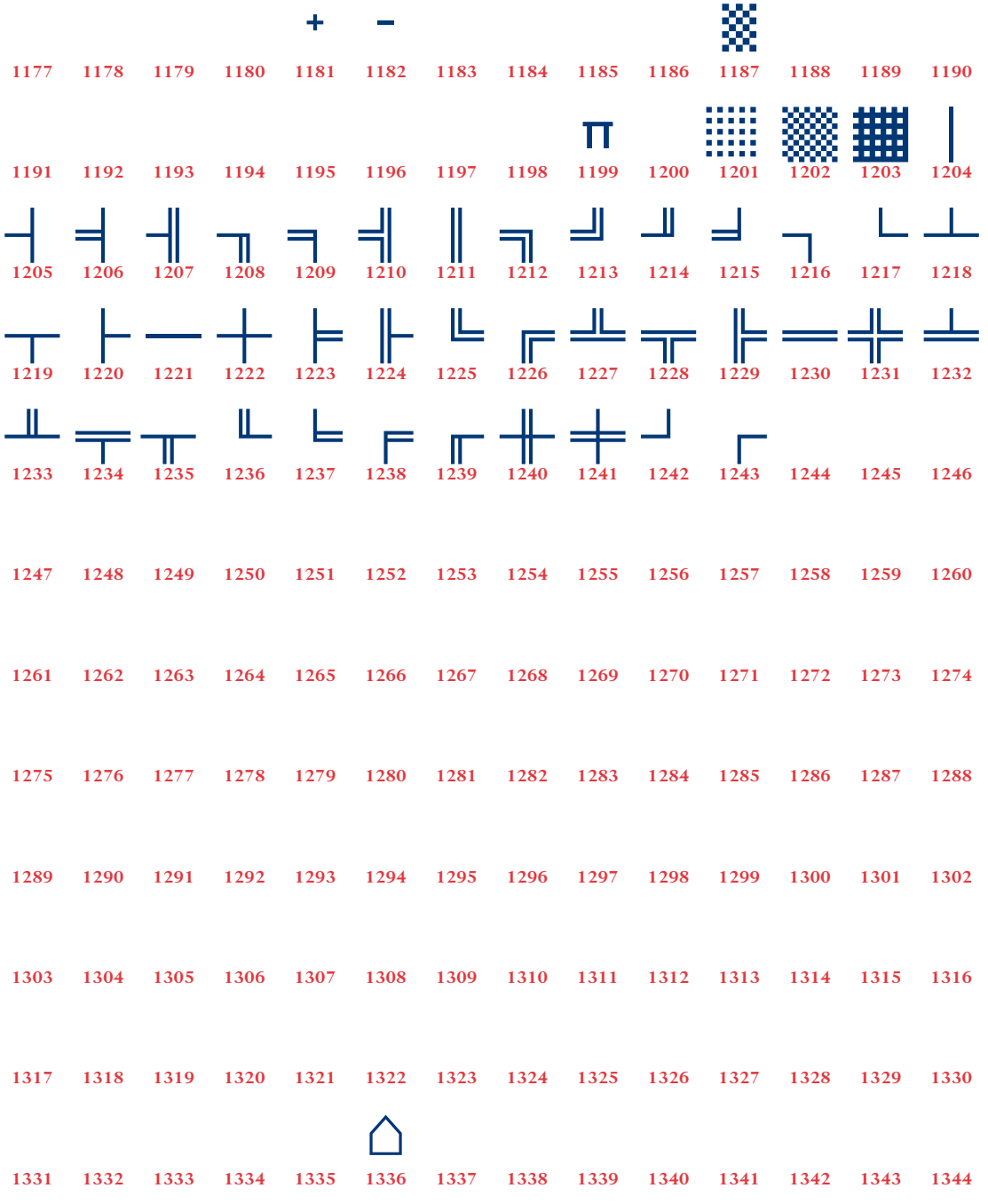
Standard character mapping for DTL FontMaster Character Layout Files.

841	842	843	844	845	846	847	848	849	850	851	852	853	854
855	856	857	858	859	860	861	862	863	864	865	866	867	868
869	870	871	872	873	874	875	876	877	878	879	880	881	882
883	884	885	886	887	888	889	890	891	892	893	894	895	896
897	898	899	900	901	902	903	904	905	906	907	908	909	910
911	912	913	914	915	916	917	918	919	920	921	922	923	924
925	926	927	928	929	930	931	932	933	934	935	936	937	938
939	940	941	942	943	944	945	946	947	948	949	950	951	952
953	954	955	956	957	958	959	960	961	962	963	964	965	966
967	968	969	970	971	972	973	974	975	976	977	978	979	980
981	982	983	984	985	986	987	988	989	990	991	992	993	994
													
995	996	997	998	999	1000	1001	1002	1003	1004	1005	1006	1007	1008

Standard character mapping for DTL FontMaster Character Layout Files.



Standard character mapping for DTL FontMaster Character Layout Files.



Standard character mapping for DTL FontMaster Character Layout Files.

1345	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1358
1359	1360	1361	1362	1363	1364	1365	1366	1367	1368	1369	1370	1371	1372
1373	1374	1375	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1386
1387	1388	1389	1390	1391	1392	1393	1394	1395	1396	1397	1398	1399	1400
1401	1402	1403	1404	1405	1406	1407	1408	1409	1410	1411	1412	1413	1414
1415	1416	1417	1418	1419	1420	1421	1422	1423	1424	1425	1426	1427	1428
1429	1430	1431	1432	1433	1434	1435	1436	1437	1438	1439	1440	1441	1442
1443	1444	1445	1446	1447	1448	1449	1450	1451	1452	1453	1454	1455	1456
1457	1458	1459	1460	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470
1471	1472	1473	1474	1475	1476	1477	1478	1479	1480	1481	1482	1483	1484
1485	1486	1487	1488	1489	1490	1491	1492	1493	1494	1495	1496	1497	1498
1499	1500	1501	1502	1503	1504	1505	1506	1507	1508	1509	1510	1511	1512

Standard character mapping for DTL FontMaster Character Layout Files.

2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030
2031	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044
2045	2046	2047	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058
2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072
2073	2074	2075	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086
2087	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2099	2100
Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι	Κ	Λ	Μ	Ν	Ξ
2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111	2112	2113	2114
Ο	Π	Ρ	Σ	Τ	Υ	Φ	Χ	Ψ	Ω				
2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127	2128
2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142
2143	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156
2157	2158	2159	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170
2171	2172	2173	2174	2175	2176	2177	2178	2179	2180	2181	2182	2183	2184

Standard character mapping for DTL FontMaster Character Layout Files.

2185	2186	2187	2188	2189	2190	2191	2192	2193	2194	2195	2196	2197	2198
2199	2200	2201	2202	2203	2204	2205	2206	2207	2208	2209	2210	2211	2212
2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223	2224	2225	2226
2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239	2240
2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254
2255	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268
2269	2270	2271	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282
2283	2284	2285	2286	2287	2288	2289	2290	2291	2292	2293	2294	2295	2296
2297	2298	2299	2300	α 2301	β 2302	γ 2303	δ 2304	ε 2305	ζ 2306	η 2307	ϑ 2308	ι 2309	κ 2310
λ 2311	μ 2312	ν 2313	ξ 2314	ο 2315	π 2316	ρ 2317	σ 2318	τ 2319	υ 2320	2321	χ 2322	ψ 2323	ω 2324
ς 2325	2326	2327	2328	2329	2330	2331	φ 2332	2333	2334	2335	2336	2337	2338
2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351	2352

Appendix IX: Font Family IDs

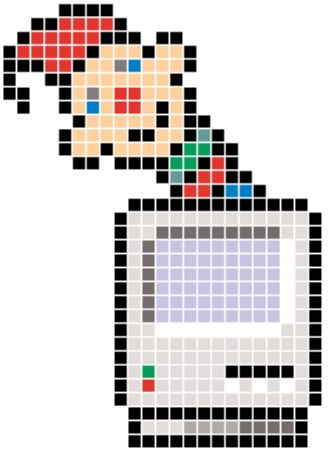
Each script inteface system that can run on the Macintosh has a range of font family IDs assigned to it. The position in the font menu of an application is determined by the ID.

Script	*Script ID*	*Font Family IDs*
System Reserved	All	0 – 1
Roman	0	2 – 16382
System Reserved	0	16383
Japanese	1	16384 – 16895
Chinese	2	16896 – 17407
Korean	3	17408 – 17919
Arabic	4	17920 – 18431
Hebrew	5	18432 – 18943
Greek	6	18944 – 19455
Russian	7	19456 – 19967
Reserved	8	19968 – 20479
Devanagari	9	20480 – 20991
Gurmukhi	10	20992 – 21503
Gujarati	11	21504 – 22015
Oriya	12	22016 – 22527
Bengali	13	22528 – 23039
Tamil	14	23040 – 23551
Telugu	15	23552 – 24063
Kannada	16	24064 – 24575
Malayalam	17	24576 – 25087
Sinhalese	18	25088 – 25599
Burmese	19	25600 – 26111
Cambodian	20	26112 – 26623
Thai	21	26624 – 27135
Laotian	22	27136 – 27647
Georgian	23	27648 – 18159
Armenian	24	28160 – 28671
Maldivian	25	28672 – 29183
Tibetan	26	29184 – 29695
Mongolian	27	29696 – 30207
Ethiopian	28	30208 – 30719
Non-Cyrillic Slavic	29	30720 – 31231
Vietnamese	30	31232 – 31743
Sindhi	31	31744 – 32255
Uninterpreted Symbols	32	32256 – 32767

DTL IconDropper is a 'drag & drop' programme for the Power Macintosh that makes it very easy to change the icons, filetype and creator of PostScript Type 1 fonts. This can be done in batch by simply dropping multiple PS Type 1 fonts on DTL IconDropper.

1. *Preparations*

Place a template file that is named «t1_icons.temp» in the directory where the programme is located. This file must contain all resources which have to be present in the final PS Type 1 font. Usually these are BNDL, FREF, ICN# and the signature resource with the same name as the creator of the file. Maybe, you will also add icl4, icl8, ics4, ics8 and ics#. Please note that there must be no POST resources in the template! Filetype and creator of the template file have to be set appropriately. As a basis the supplied 't1_icons.temp' file can be used for customizing in ResEdit.



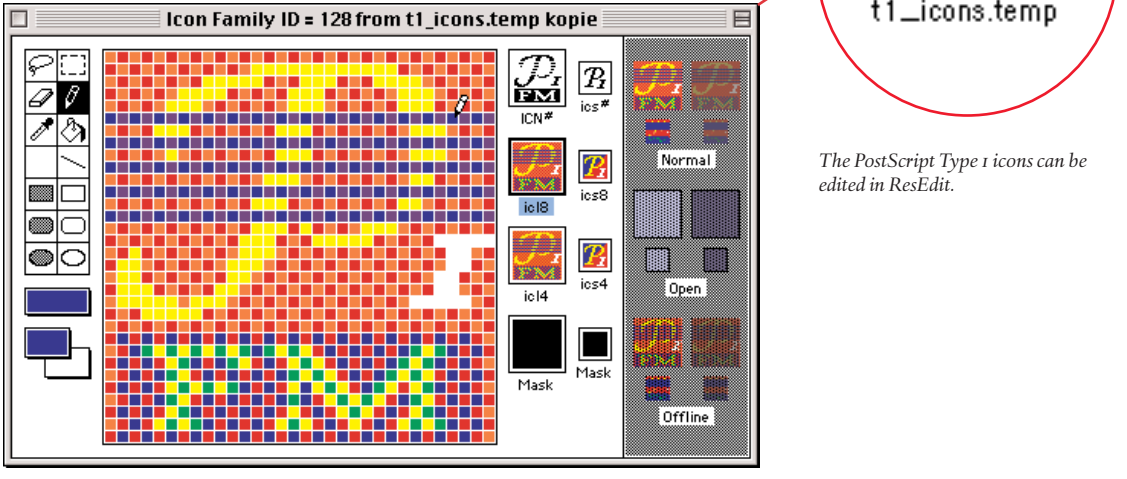
ResEdit can be used to customize the 't1_icons.temp' file or any other font file that will be used as a basis.

2. *Using the programme*

Select all the PS Type 1 fonts which you want to change and drag them onto the DTL IconDropper programme. The programme will open all files and remove all resources with exception of the POST resources. Successively it will copy all resources from the template file to the PostScript Type 1 fonts. Finally it will change filetype and creator to those of the template file.

3. *Troubleshooting*

In case you get a blank icon after changing the creator, the problem is probably solved by rebuilding the desktop.



The PostScript Type 1 icons can be edited in ResEdit.

1. Windows and Mac OS

1.1 *Invalid or missing Character Layout File*

Sometimes a DTL FontMaster module will notify you that the Character Layout File 'beeditor.cha' is invalid or missing. Probably because there is a wrong preference file on your system. In this case just click 'OK' and start the programme. Then you can locate the Character Layout File 'beeditor.cha' in the directory where you installed DTL FontMaster (for instance in: C:\Program Files\Dutch Type Library\DTL FontMaster [version#]\). You have to do this once; next time the programme will locate the 'beeditor.cha' automatically.

1.2 *Corrupted preference file*

In case a module of DTL FontMaster under Windows or Mac OS has difficulties to locate a file, the preference file is probably corrupted and has to be removed.

1.3 *UFM file compatibility problem*

UFM files which are generated on the Mac are not always recognized on the PC. This is quite a tricky problem because DTL DataMaster will generate a new UFM file that will overwrite the original UFM file when the 'Advanced' section is opened from the 'Export Fonts' dialog and the 'OK' button is pushed here. This way all the information in the original UFM file will be lost. The problem is simply to solve by opening the original UFM file in WordPad and to save it as a text file; DTL DataMaster will recognize it then.

1.4 *Improper font behaviour*

Fonts generated with DTL DataMaster must contain a space because otherwise the fonts will behave improperly. This means that also fonts that contain only one or more logos must have a space defined. The nbspace is automatically generated by the programme based on the width of the space.

1.5 *Error during TrueType generation*

In case an error occurs during the generation of a TrueType font ('return code from MakeFondts DLL is 1'), this might be caused by the formatting of the AFM file which is used for adding the kerning info. A simple solution for this is to copy the content of the AFM file and to paste it in SimpleText on the Mac or in WordPad on the PC and to save it as a text file.

1.6 *Unable to import AFM file*

Converting a PostScript Type 1 font to a BE or IK database in DTL DataMaster sometimes results in the following message: 'warning: unable to copy AFM file'. This problem is easy to solve by just copying the original AFM file to the same directory where the newly generated BE or IK file is situated. Take care to rename the AFM file according to the name of the BE or IK file while retaining the .afm extension.

2. Windows

2.1 *Installer is not recognized*

In case the Windows Installer is not recognized by your 95/98 or NT system, you need to install the 'Installer for Windows Installer' first. There are two versions available: one for Windows 95/98 and one for Windows NT. These are supplied on the CD version of the DTL FontMaster Utilities also. Alternatively you can download the 'Installer for Windows Installer' from the Microsoft web site. All other Windows systems will recognize the (.msi) Windows Installer automatically.

2.2 *Opening a font database from CD*

Opening the fmdemo.be and .ik font files directly from the demo CD will result in an empty database; you have to copy the fonts to your harddisk first. The most recent installer places the fmdemo.be and .ik files also in the same directory as the DTL FontMaster program files.

2.3 *Opening a font database from CD*

Importing a font directly from the fonts directory in the Windows system using DTL DataMaster is not possible; you have to copy the font file to a different location on your harddisk first.

3. Mac OS

3.1 *Unable to start a FM module*

In case you start on the Macintosh a FM module from the 'central switch board' and nothing happens, you have run out of memory. Close all other programs and try again.

3.2 *Slow or improper Print Preview*

In case the Print Preview function of DTL BezierMaster or DTL IkarusMaster of the Mac OS version of DTL FontMaster is very slow or does not work at all, allocate more memory for the program in the info file.

GENERAL PERMANENT
D_19_13T.BE
D_19_33T.BE
GENERAL OPTIMIZED
D_19_13T.BE
D_19_33T.BE
OPENTYPE
DO19X13T.BE
DO19X33T.BE
DP19X13T.BE
DP19X33T.BE
DX19X13T.BE
DX19X33T.BE
DO19X13T.UFM
DO19X33T.AFM
DO19X13T.PAR
DO19X33T.UFM
DO19X13T.AFM
DO19X33T.PAR
DP19X13T.UFM
DP19X13T.AFM
DP19X13T.PAR
DP19X33T.UFM
DP19X33T.AFM
DP19P33T.PAR
et cetera
WEST
DO19013T.BE
DO19033T.BE
DO19C13T.BE
DO19C33T.BE
DO19813T.BE
DO19833T.BE
DO19013T.UFM
DO19013T.AFM
DO19013T.PAR
DO19C13T.UFM
DO19013T.AFM
DO19C13T.PAR
DO19813T.UFM
DO19813T.AFM
DO19813T.PAR
DO19033T.UFM
DO19033T.AFM
DO19033T.PAR
et cetera
EAST
DE19013T.BE
DE19033T.BE
DE19C13T.BE
DE19C33T.BE
DE19813T.BE
DE19833T.BE
DE19013T.UFM
DE19013T.AFM
DE19013T.PAR
DE19C13T.UFM
DE19013T.AFM
DE19C13T.PAR
DE19813T.UFM
DE19813T.AFM
DE19813T.PAR
DE19033T.UFM
DE19033T.AFM
DE19033T.PAR
et cetera
TURKISH
DT19013T.BE
DT19033T.BE
DT19C13T.BE
DT19C33T.BE
DT19813T.BE
DT19833T.BE
DT19013T.UFM
DT19013T.AFM
DT19013T.PAR
DT19C13T.UFM
DT19013T.AFM
DT19C13T.PAR
DT19813T.UFM
DT19813T.AFM
DT19813T.PAR
DT19033T.UFM
DT19033T.AFM
DT19033T.PAR
et cetera
CYRILLIC
DC19013T.BE
DC19033T.BE
DC19C13T.BE
DC19C33T.BE
DC19813T.BE
DC19833T.BE
DC19013T.UFM
DC19013T.AFM
DC19013T.PAR
DC19C13T.UFM
DC19013T.AFM
DC19C13T.PAR
DC19813T.UFM
DC19813T.AFM
DC19813T.PAR
DC19033T.UFM
DC19033T.AFM
DC19033T.PAR
et cetera
GREEK
DG19013T.BE
DG19033T.BE
DG19C13T.BE
DG19C33T.BE
DG19813T.BE
DG19833T.BE
DG19013T.UFM
DG19013T.AFM
DG19013T.PAR
DG19C13T.UFM
DG19013T.AFM
DG19C13T.PAR
DG19813T.UFM
DG19813T.AFM
DG19813T.PAR
DG19033T.UFM
DG19033T.AFM

All the files generated by the FM modules are platform-independent. At the Dutch Type Library the font production files are centralized on a server and can be reached from any Mac or PC in the network. Because of the fact that the FM data is platform-independent, no conversion has to be done to enable editing of the files under Mac OS or Windows.

The font data consist of five files: the collection of glyphs stored in the BE or IK format, the UFM file that contains the naming, copyright, ID and some conversion information, the AFM and FEA files, and the PAR file. Normally a UFM file is made once and (almost) never changed again. As many of the supporting files of FM, also the UFM file is a simple text file which can be altered directly using a text editor. The PAR file contains path information to the kerning data, which can be stored anywhere.

The BE/IK, UFM and PAR files have to be located in the same directory and are automatically 'connected' by name. If a BE or IK database is selected in DTL DataMaster the font naming (and ID info) is taken from the UFM file with the same name. This system is quite versatile.

At DTL the databases are stored in an old fashion way with an eight character code. The database of DTL Caspari Regular for instance is named C_94_13T. The underscores are replaced with respectively the character that indicates the code page and what we call the 'Standard' (O) and 'Special' (C) info. The last character indicates text (T), display (D) or Poster (P). C_94 is the code for DTL Caspari and 13 means Regular (33 is Italic, 14 is Medium, 34 is Medium Italic, etc.). As a result C0940I3T means Regular with Western European lay-out and tabular figures, and CE94CI3T is the code for Eastern European with old style figures. CP94XI3T is the code for the OpenType Pro font.

All glyphs of one weight/style are stored in one 'master' database. Copies of these masters, which are checked and corrected first with DTL ContourMaster, are together with the appropriate UFM and PAR files stored in different directories for Western- and Eastern European, Cyrillic, Greek, etc. This makes batch generation in DTL DataMaster very easy.

The databases for for instance Western- and Eastern European, Cyrillic and Greek are stored respectively as C0940I3T, CE940I3T, CC940I3T and CG940I3T. The second zero is replaced by a C in case of the versions with the old style figures. The appropiate UFM and PAR files should have the same naming as the corresponding databases to secure automatic connection to these databases.

The only thing that has to be taken care of when a (series of) font(s) has to be generated, is the selection of the appropriate Character Layout file (.cha) that contains the required code page.

Please note that the generation of Mac fonts is only possible with the Mac OS version of DTL DataMaster. In mixed environments it is enough to have the editors for both platforms but only the Mac version of DTL DataMaster for generating different font formats.

Fontographer versus DTL FontMaster

Designers who used to work with Fontographer in the past may perhaps sometimes find it difficult to locate the same functions in DTL FontMaster. Here is a short list of functions with the shortcuts in Fontographer versions 3.5 and 4.x for Mac OS and version 3.5 for Windows.

Fontographer uses three different names to label the relation between the Bezier Control points (BCPs) and the Anchor points: Curve, Corner and Tangent point.

The Curve point is what is called a Smooth Anchor point in DTL FontMaster and both BCPs are placed on a virtual line that intersects the anchor point; moving one of the BCPs will influence the position of the other BCP automatically. The shortcut for this point is in Fontographer 3.5 <Command> + 5 on the Mac and <Ctrl> + 5 on the PC and <Command> + 8 in Fontographer 4.x on the Mac.

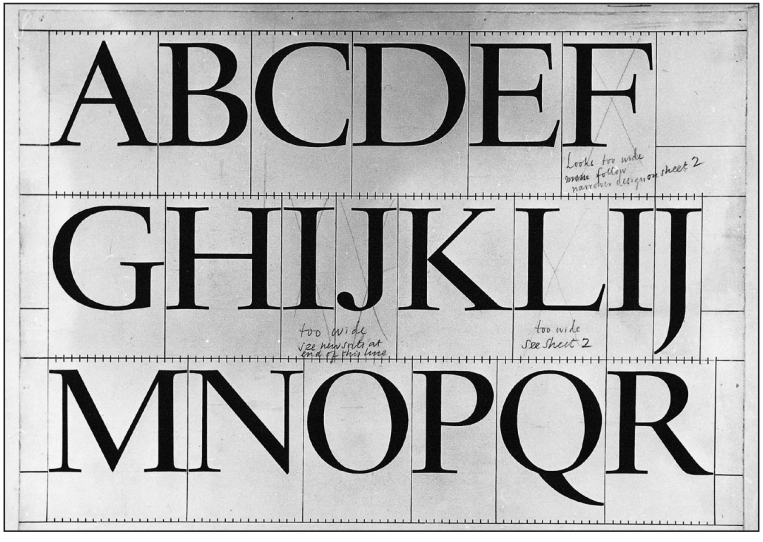
The corner point in Fontographer is called Anchor point in DTL FontMaster. The BCPs of such a point can be moved completely independent from each other. The shortcuts in Fontographer 3.5 for Mac OS and Windows are respectively <Command> + 4 and <Ctrl> + 4 and <Command> + 9 for Fontographer 4.x on the Mac.

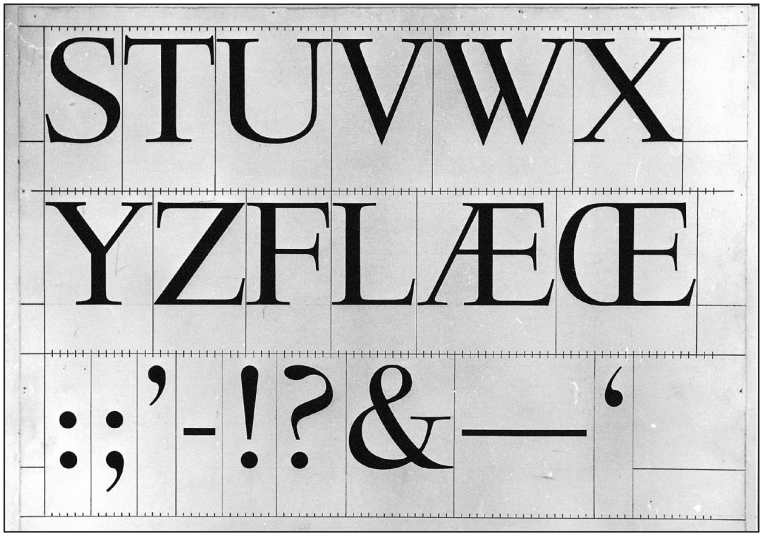
To force tangent continuity between adjacent sections in Fontographer a so called Tangent point is used. In DTL FontMaster a Smooth Anchor point is used for this purpose and in case the adjacent section is a straight line, tangent continuity will be forced automatically. The shortcuts are <Command> + 3 and <Ctrl> + 3 in Fontographer 3.5 respectively on the Mac and in Windows. In Fontographer 4.x on the Mac <Command> + 0 is used.

Changing points in DTL FontMaster is very easy: just press the <Ctrl> key and mouse-click on the point. This works on both Mac and PC. Bezier Control points can be removed by simply selecting these and subsequently pressing the <backspace> key. To lock the direction of the Bezier Control point, for which the combination <Alt> + <Shift> is used in Fontographer, just select the point in DTL FontMaster and subsequently press the <Shift> key and, while keeping this key pressed, move the Bezier Control point with the mouse.

In Fontographer 3.5 the contour can be altered only by moving the BCPs. In Fontographer 4.x the contour itself can be moved also. In DTL FontMaster the default setting is that only the BCPs can be moved. By selecting the Shift Outline tool from the Function Tool Bar only the contour can be moved.

In DTL FontMaster the contour is filled by <Command> + F on the Mac and <Ctrl> + F under Windows. The display of the points can be turned on and off by using <Command> + M (Mac) and <Ctrl> + M (Windows).





Two of the nine sheets containing the drawings Jan van Krimpen made in 1938 for Haarlemmer. (collection DTL)

DTL FontMaster *Manual*

Edition 1.0/2004; supports DTL FontMaster version 2.1.1

ISBN 90-75005-06-7

Text Frank E. Blokland and Dr. Jürgen Willrodt

Proof English read by Dr. Peter Kahrel

Design Frank E. Blokland

Typeset in DTL Argo, DTL Haarlemmer and DTL Haarlemmer Sans

Print digital, 1200 dpi; Printed at Dutch Type Library

Paper Mellotex smooth, 115 grams

Dutch Type Library

Kruisstraat 33

5211 DT 's-Hertogenbosch

The Netherlands

phone +31 (0)73 614 95 36

fax +31 (0)73 613 98 23

e-mail info@dutchtypelibrary.com

website www.dtl.nl

website www.fontmaster.nl

DTL Germany

Poppenbütteler Bogen 36

22399 Hamburg

Germany

phone +49 (0)40 60 60 52 28

fax +49 (0)40 60 60 51 11

e-mail info@dutchtypelibrary.de

website www.dutchtypelibrary.de

website www.fontmaster.de

FontMaster™ *Utilities*

Edition 1.0/2004
ISBN 90-75005-06-7