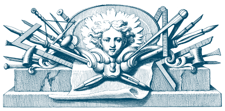
Dutch Type Library

DTL OTMaster

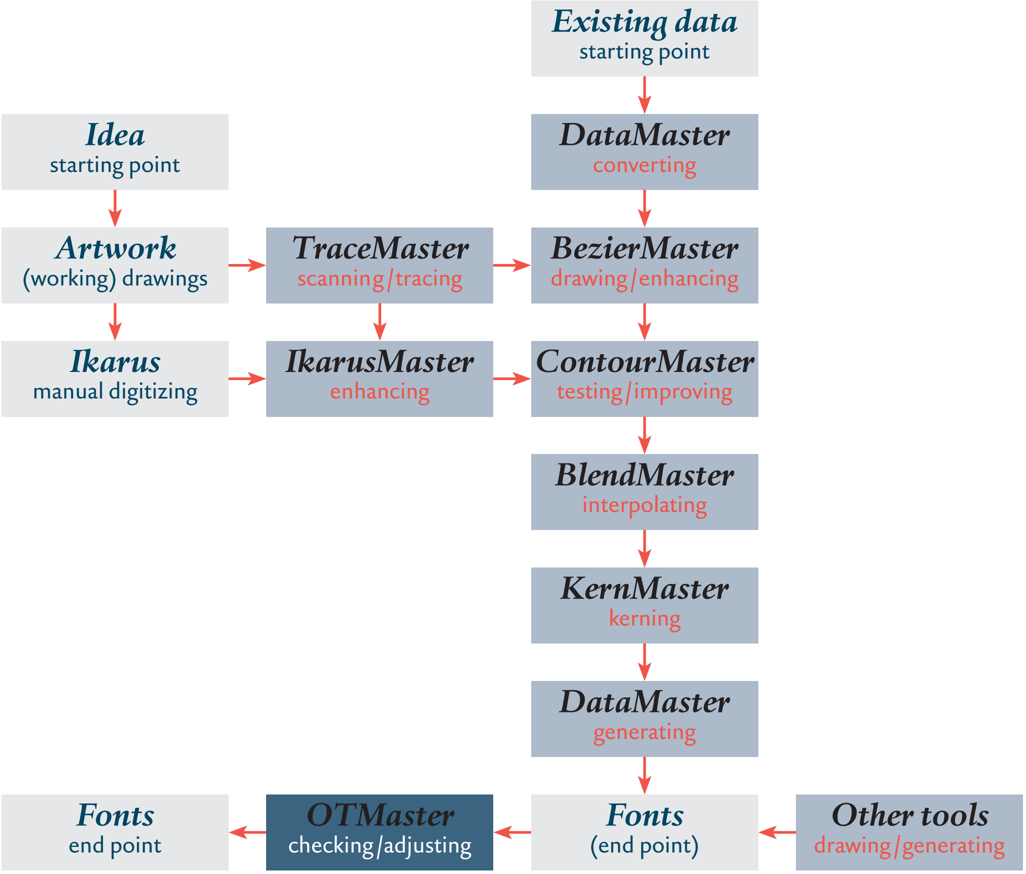


’s-Hertogenbosch / Hamburg
2009

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.



Introduction

DTL OTMaster is a stand-alone application whose interface makes it easy to review and edit .otf and .ttf fonts' tables, no matter whether these fonts have been generated with IkarusMaster, BezierMaster, DataMaster, or any other font editor.

Font editors, like the FontMaster suite, rely on their own internal data formats for type design and font production. With FontMaster, this is either an IK or a BE file, for Ikarus and Bezier outlines respectively, along with various data files for naming font and glyphs, for kerning and definition of typographic layout features. From these data, ready-to-use binary fonts are compiled as the very last step.

OTMaster is a tool whose purpose is to inspect and adjust such ready-to-use binary fonts, irrespective with which font editor they have been created. Its advantage is that it allows editing of tables in a graphic user interface. Moreover, it comes with additional tools like a Glyph Editor to proof, edit and even draw glyph outlines, a 'kern' Table Viewer to proof and refine the kern table, and a ‘GSUB’/‘GPOS’ Viewer to visually test (and in case of GPOS adjust) these OpenType layout tables.

OTMaster can open, edit and save fonts with SFNT file structure: CFF-based and TT-based OpenType fonts, TrueType fonts and TTC (TrueType Collection) fonts.

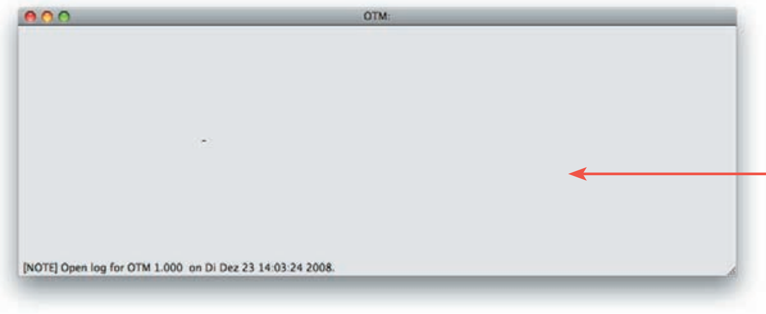
Because OTMaster enables you to edit a binary font’s tables, which can be compared to open-heart surgery, it is highly recommended that you know the OpenType specification, at least as regards the tables whose entries you plan to adjust. ►*local* ►*www* (The current version is 1.5. Please note that there were minor changes, e.g. the OS/2 table has been updated to version 4, and nameIDs have been added to the name table.)

About This Manual

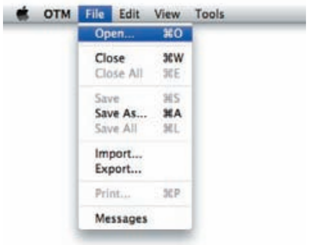
There are a number of links in this document. The first type of link is ‘►*local*’ and links to the OpenType specification which Microsoft kindly provided so as to accompany this manual. As long as the folder ‘OTM Manual resources’ is located next to the manual’s PDF file, clicking on such a link will open the according local html page in your web browser, offline. The second type of link is ‘►*www*’ and will open the original online resource. This may be the preferred choice because these data may be more up-to-date. And finally there are internal links like ‘►*Chapter*’, leading from one chapter to another one.

First Steps into OTMaster

When launching OTMaster, the main dialog is empty. Since OTMaster is meant for editing existing fonts, the first step is to open a font.



```
[NOTE] Open log for OTM 1.000  on Di Dez 23 14:03:24 2008.
```



Drag & drop a font onto OTMaster's main window to open it.

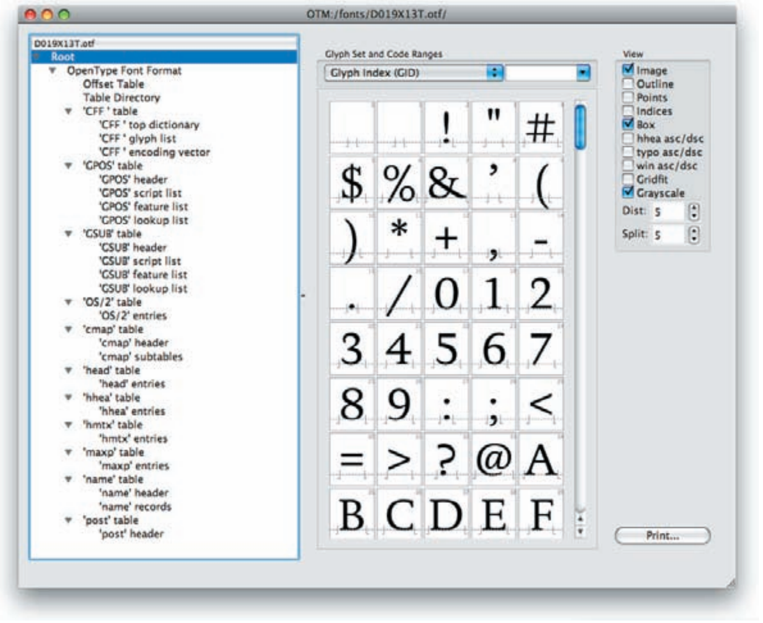
FILE MENU

Open (⌘+O) (CTRL+O)

This opens an existing font using the standard Open dialog.

Alternatively, drag & drop a font file's icon onto OTMaster's main dialog. Or drag & drop one or more font files onto the OTMaster icon to launch OTMaster and open the fonts.

*Tip: It is highly recommended that you only edit copies of fonts with OTMaster, to prevent that you inadvertedly overwrite a font when using **Save** while editing. Either, make a copy of the font file, and edit the copy in OTMaster. Or, **Save As…** a font immediately after opening it, by another name.*



Opening a font will show the table overview to the left, and the content of a selected table to the right.

Close (⌘+W) (CTRL+W)

Closes the currently active font. If you have made any changes which have not been saved yet, a dialog will ask whether you want to save these changes or not.

Close All (⌘+E) (CTRL+E)

Closes all open fonts. If you have made any changes to a font which has not been saved yet, you will be asked whether to save recent changes or not.

Save (⌘+S) (CTRL+S)

Saves the currently active font at its current location and thus overwrites the existing file.

Save As… (⌘+⇧+S) (CTRL+⇧+S)

Saves the currently active font but allows you to determine a new location in the standard Save dialog.

Save All (⌘+L) (CTRL+L)

All open fonts will be saved at their current locations.

Import…

This dialog's options popup currently offers the following file types:

— character layout file (.cha)

Importing a text-based .cha file will allow OTMaster to show this file's glyph names in tables' **Comment** column. This may be helpful if you have generated a CID-keyed font from FontMaster and plan to edit it in OTMaster. Note that importing a .cha file does not have any impact on the font file or any of its tables!

OTMaster accepts .cha files that map, per glyph, its glyph index, Unicode codepoint and glyph name. Please see the column to the right.

— Adobe FDK feature file

This will import – i.e. compile – selected OpenType layout tables from a .fea file which conforms to Adobe's feature file syntax as described in the *OpenType Feature File Specification.* ▶ *www*

OTMaster does not return any detailed report in case that import, i.e. compilation, fails because of mistakes in your .fea file! So please make sure that your .fea file does not contain any syntax or other errors. One common issue appears to be too many pos statements in the 'kern' feature.

```
Version 002.000
Starttable
GlyInd;UNINum;PSName
0;;.notdef
1;x0020;space
2;x0021;exclam
3;x0022;quotedbl
4;x0023;numbersign

Endtable
```

*A .cha file as exported by OTMaster and accepted for import. The header must read '**Version 002.00**'. Between tags '**Starttable**' and '**Endtable**', a semicolon-separated table holds the glyph information. The table's header indicates that each glyph is identified by glyph index (**GlyInd**) and optionally is provided with a Unicode codepoint (**UNINum**) and a glyph name (**PSName**).*

The order of columns is arbitrary, but it is important that the order of data (in each row) matches the order determined in the table header.

*OTMaster .cha files differ from those employed in FontMaster. OTMaster identifies glyphs by glyph index (**GlyInd**) while FontMaster identifies glyphs by a URW number (**URWNum**). To exchange .cha files between OTMaster and FontMaster, just replace the keyword '**GlyInd**' by the keyword '**URWNum**' (or the other way round). Please note that FontMaster .cha files may hold further information, in additional columns headed by keywords as described in the FontMaster manual.*

*Being semicolon-separated, a .cha file can easily be edited e.g. in Excel: Add the suffix '**.csv**' to the .cha file's file name and in Excel's Open dialog select 'Text (*.prn; *.txt; *.csv)' as file type. After adjusting the table and saving it, remove the '**.csv**' suffix from the file name.*

Export …

Currently, these data may be exported:

— character layout file (.cha)

A .cha file as exported by OTMaster maps, per glyph, its glyph index, Unicode codepoint and glyph name.

An OpenType font's .cha file is helpful when importing this font in FontMaster: if you replace 'GlyInd' by 'URWNum' in the .cha file's header, the original glyph indices will serve as URW numbers which identify glyphs in the FontMaster suite.

— Adobe FDK feature file

This will export selected OpenType layout tables as a .fea file in Adobe's feature file syntax. Export means that the tables will be dumped which involves interpretation as described in the note. You are advised to read and possibly correct exported feature files, especially if a font's layout behavior is complex: OTMaster's AFDKO-syntax dump is not perfect yet.

Please consult Adobe's *OpenType Feature File Specification* for details about the feature file keywords, syntax and examples. ▸ *www*

Note: Feature files usually identify glyphs by their glyph names. This however means that if you 'transfer' typographic layout features from one of your fonts to another one, glyphs in both fonts need to share identical names, and all glyphs referenced in the feature file must be present in the destination font.

*Please mind that such a 'transfer' is not lossless because it involves interpretation: 1. **Export** will dump OpenType layout tables into an AFDKO-syntax feature file, and 2. **Import** will compile OpenType layout tables from a feature file.*

Also, 'transferring' features from one font to another often is not that rewarding because most likely each font has a different glyph set with different alternates, so that different feature behavior is required.

Print … (⌘+P) (CTRL+P)

Because it does not make much sense to print data as represented in the user interface, it is recommended that you switch, in the **View** menu, to **Text Dump** mode and save either the text dump or XML files and print this.

Preferences (⌘+,) (CTRL+,)

Please see the ▸ *Preferences* chapter.

*Note: In the Mac OS version of OTMaster, both **Preferences** and **Quit** are located in the OTM menu.*

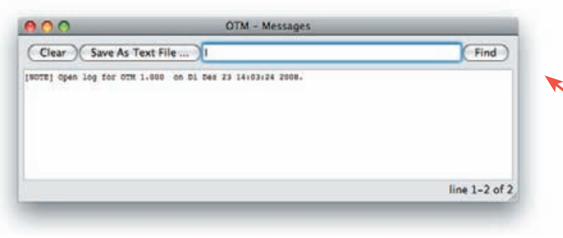
Quit (⌘+Q) (CTRL+Q)

This will quit the application. If fonts have been changed but not saved yet, you will be asked whether you want to save these changes or not.

Messages

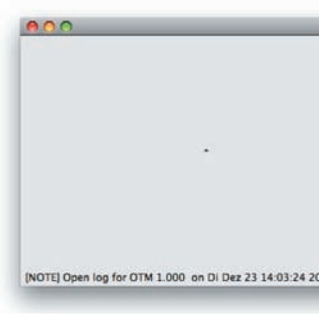
This opens the Messages window which collects all status messages as shown at the bottom of the OTMaster's main dialog.

You may **Clear** the panel, **Save As Text File …** its content, and search in it with help of **Find**.



```
[NOTE] Open log for OTM 1.000  on Di Dez 23 14:03:24 2008.
```

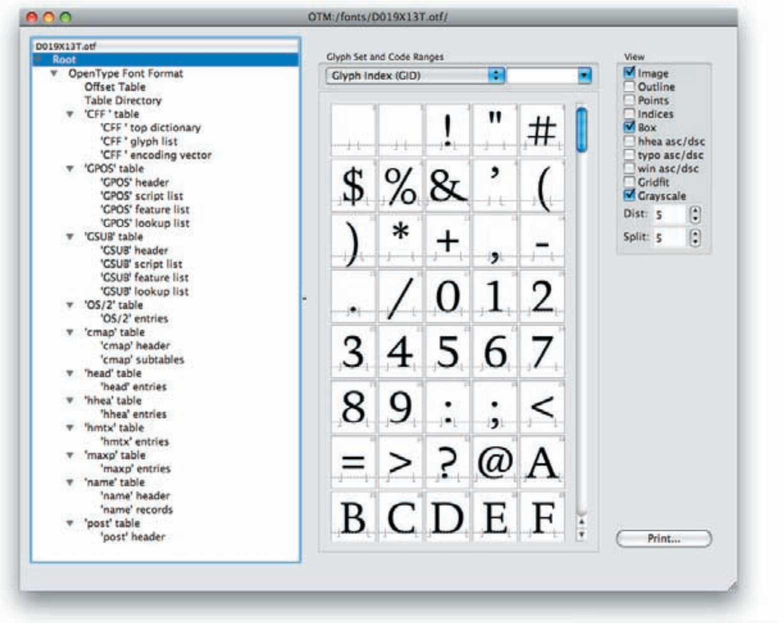
A typical status message. All of them will be collected in the Messages window.



```
[NOTE] Open log for OTM 1.000  on Di Dez 23 14:03:24 2
```

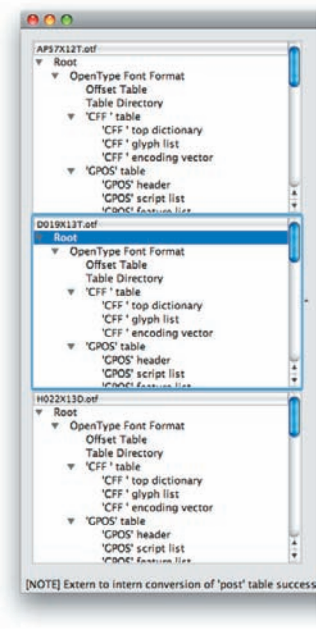
Inspecting Tables

Once a font is opened, OTMaster's main dialog consists of two areas:



☑ Image
☐ Outline
☐ Points
☐ Indices
☑ Box
☐ hhea asc/dsc
☐ typo asc/dsc
☐ win asc/dsc
☐ Gridfit
☑ Grayscale
Dist: 5
Split: 5

OTMaster's main window showing a CFF*-based OpenType font. If more than one font is opened, the table overview is tiled into as many segments as there are open fonts. The currently active font's segment 'glows' in the Mac* OS *version:*



[NOTE] Extern to intern conversion of 'post' table success

The leftside area presents a table overview with the font's file name in a header. If more than one font has been opened, this area consists of as many segments as there are open fonts. The rightside area shows the entries of the table which is currently selected in a font's table overview.

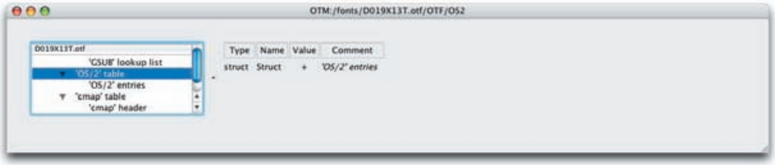
The table overview usually starts with the Root entry which represents the font as a whole. Root is the default selection in the table overview, and accordingly the rightside content area presents this font's glyph set. Strictly speaking, this glyph set overview is merely for your convenience – Root does not represent any specific table. Functionality and options for Root glyph overview and Font Viewer are identical and are described in the chapter ▶ *Font Viewer*.

The term OpenType Font Format *refers to both* CFF*-based and* TT*-based OpenType fonts. The former store outline data in a* CFF *table, the latter store them in a* glyf *table.*

Root includes OpenType Font Format which indicates that a font is an OpenType font. This in turn is populated with the OpenType font's data according to SFNT file structure: An Offset Table tells how many tables this font contains, plus information for binary search. The Table Directory points out start (offset) and length of every table. These then are followed by all tables present in the font. (A TTC font would indicate TrueType Collection Format rather than OpenType Font Format. The overall structure would differ slightly too. Please consult the chapter ▶ *TTC Fonts.*)

Offset Table and Table Directory are not supposed to be adjusted manually. They are read-only and will be updated automatically as soon as tables are modified, removed or added.

Simple tables include only [table] entries. Selecting [table] table in the table overview will exhibit the top level of this table in the content area.



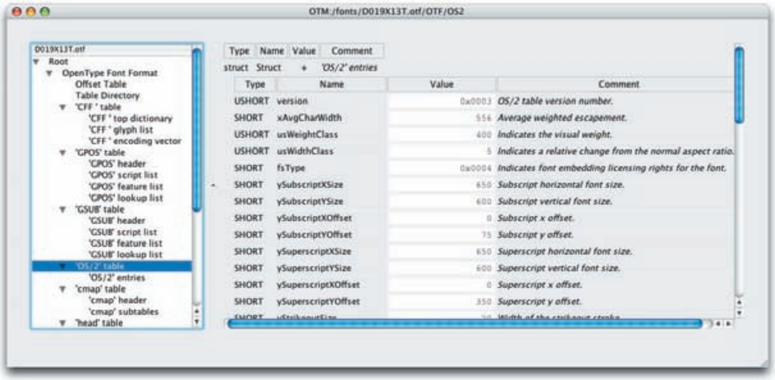
OTM:/fonts/D019X13T.otf/OTF/OS2

- D019X13T.otf
    - 'GSUB' lookup list
  - ▼ 'OS/2' table
    - 'OS/2' entries
  - ▼ 'cmap' table
    - 'cmap' header

Type	Name	Value	Comment
struct	Struct	+	*'OS/2' entries*

A table's top level.

The **+** sign in the **Value** column indicates that there is more information, and a click on the **+** will fold out a nested table whose content is indented:



OTM:/fonts/D019X13T.otf/OTF/OS2

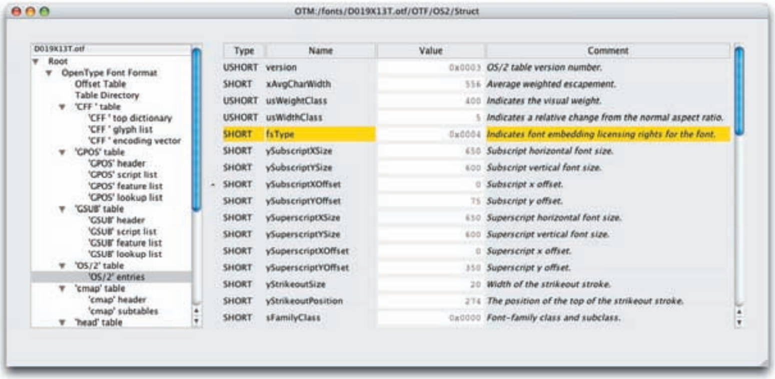
- D019X13T.otf
- ▼ Root
  - ▼ OpenType Font Format
    - Offset Table
    - Table Directory
    - ▼ 'CFF ' table
      - 'CFF ' top dictionary
      - 'CFF ' glyph list
      - 'CFF ' encoding vector
    - ▼ 'GPOS' table
      - 'GPOS' header
      - 'GPOS' script list
      - 'GPOS' feature list
      - 'GPOS' lookup list
    - ▼ 'GSUB' table
      - 'GSUB' header
      - 'GSUB' script list
      - 'GSUB' feature list
      - 'GSUB' lookup list
    - ▼ 'OS/2' table
      - 'OS/2' entries
    - ▼ 'cmap' table
      - 'cmap' header
      - 'cmap' subtables
    - ▼ 'head' table

Type	Name	Value	Comment
struct	Struct	+	*'OS/2' entries*

Type	Name	Value	Comment
USHORT	version	0x0003	*OS/2 table version number.*
SHORT	xAvgCharWidth	556	*Average weighted escapement.*
USHORT	usWeightClass	400	*Indicates the visual weight.*
USHORT	usWidthClass	5	*Indicates a relative change from the normal aspect ratio.*
SHORT	fsType	0x0004	*Indicates font embedding licensing rights for the font.*
SHORT	ySubscriptXSize	650	*Subscript horizontal font size.*
SHORT	ySubscriptYSize	600	*Subscript vertical font size.*
SHORT	ySubscriptXOffset	0	*Subscript x offset.*
SHORT	ySubscriptYOffset	75	*Subscript y offset.*
SHORT	ySuperscriptXSize	650	*Superscript horizontal font size.*
SHORT	ySuperscriptYSize	600	*Superscript vertical font size.*
SHORT	ySuperscriptXOffset	0	*Superscript x offset.*
SHORT	ySuperscriptYOffset	350	*Superscript y offset.*

Accessing a table's entries via its top level – by unfolding a nested table.

Since the top level of a table is not very informative, most of the time you may want to select [table] entries to access the table's entries directly, ready to be studied or adjusted:



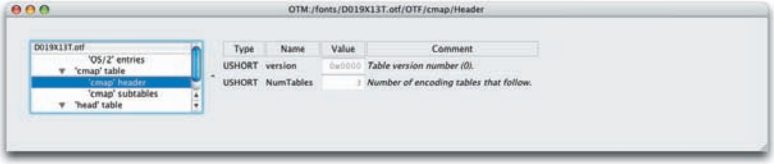
OTM:/fonts/D019X13T.otf/OTF/OS2/Struct

- D019X13T.otf
- ▼ Root
  - ▼ OpenType Font Format
    - Offset Table
    - Table Directory
    - ▼ 'CFF ' table
      - 'CFF ' top dictionary
      - 'CFF ' glyph list
      - 'CFF ' encoding vector
    - ▼ 'GPOS' table
      - 'GPOS' header
      - 'GPOS' script list
      - 'GPOS' feature list
      - 'GPOS' lookup list
    - ▼ 'GSUB' table
      - 'GSUB' header
      - 'GSUB' script list
      - 'GSUB' feature list
      - 'GSUB' lookup list
    - ▼ 'OS/2' table
      - 'OS/2' entries
    - ▼ 'cmap' table
      - 'cmap' header
      - 'cmap' subtables
    - ▼ 'head' table

Type	Name	Value	Comment
USHORT	version	0x0003	*OS/2 table version number.*
SHORT	xAvgCharWidth	556	*Average weighted escapement.*
USHORT	usWeightClass	400	*Indicates the visual weight.*
USHORT	usWidthClass	5	*Indicates a relative change from the normal aspect ratio.*
SHORT	fsType	0x0004	*Indicates font embedding licensing rights for the font.*
SHORT	ySubscriptXSize	650	*Subscript horizontal font size.*
SHORT	ySubscriptYSize	600	*Subscript vertical font size.*
SHORT	ySubscriptXOffset	0	*Subscript x offset.*
SHORT	ySubscriptYOffset	75	*Subscript y offset.*
SHORT	ySuperscriptXSize	650	*Superscript horizontal font size.*
SHORT	ySuperscriptYSize	600	*Superscript vertical font size.*
SHORT	ySuperscriptXOffset	0	*Superscript x offset.*
SHORT	ySuperscriptYOffset	350	*Superscript y offset.*
SHORT	yStrikeoutSize	20	*Width of the strikeout stroke.*
SHORT	yStrikeoutPosition	274	*The position of the top of the strikeout stroke.*
SHORT	sFamilyClass	0x0000	*Font-family class and subclass.*

Accessing a table's entries directly.

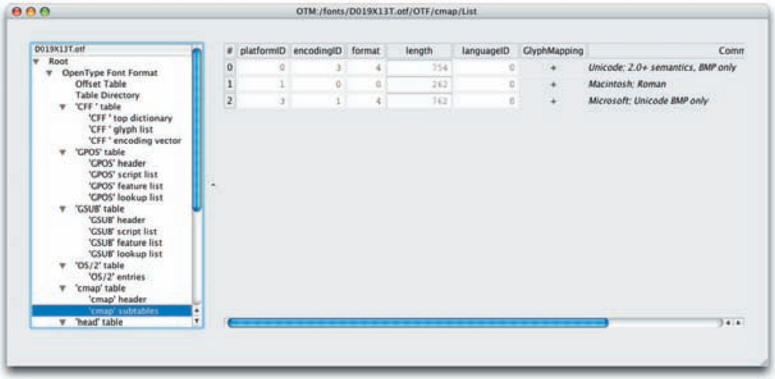
Tables of variable length which consists of one or more subtables like cmap or kern, or multiple entries like name, then [table] table will include both [table] header and [table] entries. Like Offset Table and Table Directory, the [table] header is read-only:



Type	Name	Value	Comment
USHORT	version	0x0000	*Table version number (0).*
USHORT	NumTables	3	*Number of encoding tables that follow.*

Header of a table which contains multiple subtables.

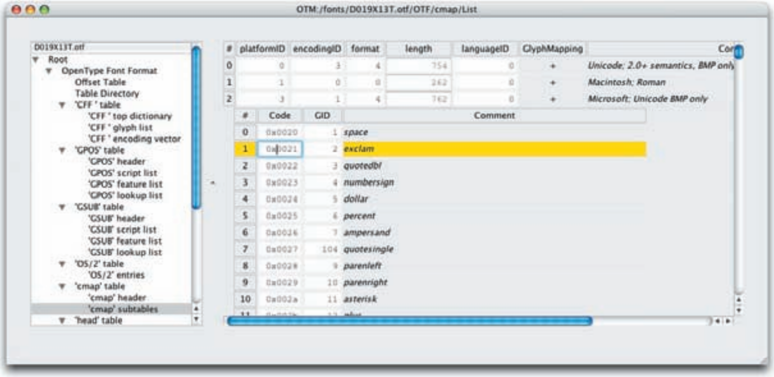
The cmap table is a good example for a table which consists of multiple subtables. This is reflected in the 'cmap' entries table content area:



#	platformID	encodingID	format	length	languageID	GlyphMapping	Comm
0	0	3	4	754	0	+	*Unicode; 2.0+ semantics, BMP only*
1	1	0	0	262	0	+	*Macintosh; Roman*
2	3	1	4	762	0	+	*Microsoft; Unicode BMP only*

A table's 'table' entries *shows the available subtables in the content area, ready to be unfolded.*

And again the **+** (in the **GlyphMapping** column) signals that per subtable there is some content to be unfolded. Clicking on one of them will reveal the respective subtable's mappings of Unicode codepoints (**Code**) to glyph indices (**GID**). In our example, we click the last subtable's **+** to see it's entries. (Clicking the **+** again will hide it.)

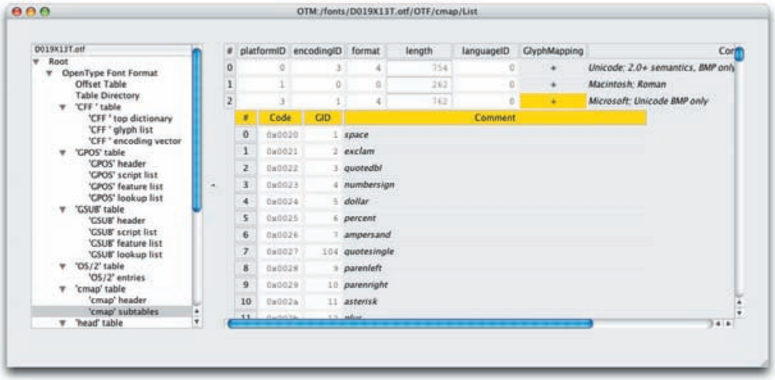


#	platformID	encodingID	format	length	languageID	GlyphMapping	Con
0	0	3	4	754	0	+	*Unicode; 2.0+ semantics, BMP only*
1	1	0	0	262	0	+	*Macintosh; Roman*
2	3	1	4	762	0	+	*Microsoft; Unicode BMP only*

#	Code	GID	Comment
0	0x0020	1	*space*
1	0x0021	2	*exclam*
2	0x0022	3	*quotedbl*
3	0x0023	4	*numbersign*
4	0x0024	5	*dollar*
5	0x0025	6	*percent*
6	0x0026	7	*ampersand*
7	0x0027	104	*quotesingle*
8	0x0028	9	*parenleft*
9	0x0029	10	*parenright*
10	0x002a	11	*asterisk*

Unfold a subtable by clicking the **+** *sign and review or edit entries. The subtable's entries are indented.*

However, some tables like GSUB or GPOS have a rather complex structure, which makes it hard to keep track of the hierarchy and the level to which displayed entries belong. To address this, and to help comparing tables of different fonts, OTMaster can display a table's or nested table's content in a separate window. Just double-click onto a nested table's header, like the one selected (and thus colored yellow) in the image below



OTM:/fonts/D019X13T.otf/OTF/cmap/List

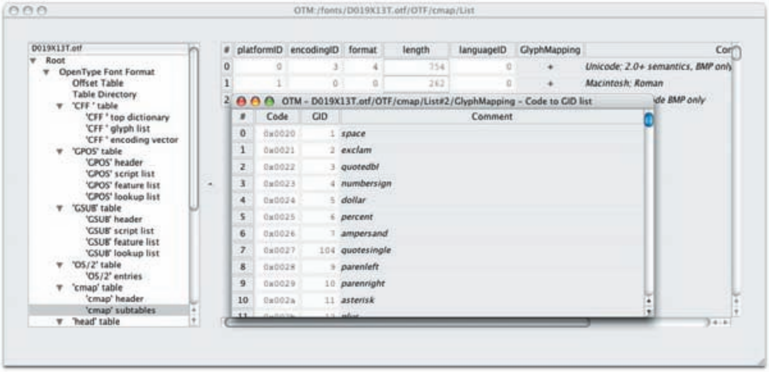
D019X13T.otf
- Root
  - OpenType Font Format
    - Offset Table
    - Table Directory
    - 'CFF ' table
      - 'CFF ' top dictionary
      - 'CFF ' glyph list
      - 'CFF ' encoding vector
    - 'GPOS' table
      - 'GPOS' header
      - 'GPOS' script list
      - 'GPOS' feature list
      - 'GPOS' lookup list
    - 'GSUB' table
      - 'GSUB' header
      - 'GSUB' script list
      - 'GSUB' feature list
      - 'GSUB' lookup list
    - 'OS/2' table
      - 'OS/2' entries
    - 'cmap' table
      - 'cmap' header
      - 'cmap' subtables
    - 'head' table

#	platformID	encodingID	format	length	languageID	GlyphMapping	Cor
0	0	3	4	754	0	+	Unicode; 2.0+ semantics, BMP only
1	1	0	0	262	0	+	Macintosh; Roman
2	3	1	4	762	0	+	Microsoft; Unicode BMP only

#	Code	GID	Comment
0	0x0020	1	space
1	0x0021	2	exclam
2	0x0022	3	quotedbl
3	0x0023	4	numbersign
4	0x0024	5	dollar
5	0x0025	6	percent
6	0x0026	7	ampersand
7	0x0027	104	quotesingle
8	0x0028	9	parenleft
9	0x0029	10	parenright
10	0x002a	11	asterisk

Click onto a nested table's header …

to create a new window. This way, you can compare multiple font's tables, and as many as you like. The only limitation is the size of your screen!



OTM:/fonts/D019X13T.otf/OTF/cmap/List

D019X13T.otf
- Root
  - OpenType Font Format
    - Offset Table
    - Table Directory
    - 'CFF ' table
      - 'CFF ' top dictionary
      - 'CFF ' glyph list
      - 'CFF ' encoding vector
    - 'GPOS' table
      - 'GPOS' header
      - 'GPOS' script list
      - 'GPOS' feature list
      - 'GPOS' lookup list
    - 'GSUB' table
      - 'GSUB' header
      - 'GSUB' script list
      - 'GSUB' feature list
      - 'GSUB' lookup list
    - 'OS/2' table
      - 'OS/2' entries
    - 'cmap' table
      - 'cmap' header
      - 'cmap' subtables
    - 'head' table

#	platformID	encodingID	format	length	languageID	GlyphMapping	Cor
0	0	3	4	754	0	+	Unicode; 2.0+ semantics, BMP only
1	1	0	0	262	0	+	Macintosh; Roman
2							de BMP only

OTM - D019X13T.otf/OTF/cmap/List#2/GlyphMapping - Code to GID list

#	Code	GID	Comment
0	0x0020	1	space
1	0x0021	2	exclam
2	0x0022	3	quotedbl
3	0x0023	4	numbersign
4	0x0024	5	dollar
5	0x0025	6	percent
6	0x0026	7	ampersand
7	0x0027	104	quotesingle
8	0x0028	9	parenleft
9	0x0029	10	parenright
10	0x002a	11	asterisk

… to open a new window for this nested table.

We have already seen some important OTMaster actions:

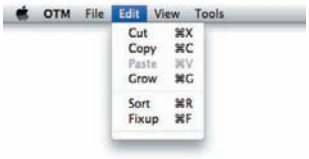
1. Click a **+** sign to show or hide a nested table.
2. Change a table's entries in usual textboxes.
3. Double-click a nested table's header to show its content in a new window.

And one addition:

4. Switch from column to column with ⇥ (to right) and ⇧+⇥ (or ⇤; to left) and from line to line with ↑ (arrow up) and ↓ (arrow down) keys.

Editing Tables

The most important functions for removing or adding data are found in the **Edit** menu. All of these functions apply to individual tables selected in the table overview, or table entries selected in the table content area.



EDIT MENU

Cut [and Delete] (⌘+X) (CTRL+X)

Cuts a selected table or table entry. This will remove the table from the font, or entry from the table, and keep it in the clipboard – this means, Cut is equivalent to **Delete**.

*Note: There is no special **Delete** function in OTMaster. However, **Cut** serves the same purpose.*

Copy (⌘+C) (CTRL+C)

Copies a selected table or table entry into the clipboard without removing it from the font.

Paste (⌘+V) (CTRL+V)

Pastes a table from the clipboard into the currently selected font, or pastes a table entry from the clipboard into the currently selected table.

Grow (⌘+G) (CTRL+G)

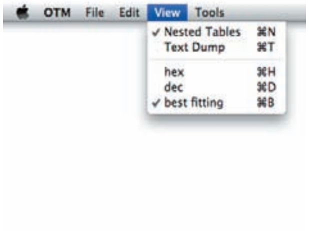
This function does exactly what it says – help growing a table. In tables of variable length (such as name, cmap, kern) select an entry (like a name record in the name table) and use Grow to duplicate it. In other tables, Grow will create a new, empty entry.

Sort (⌘+R) (CTRL+R)

Sorts table entries in tables of variable length. For example, this will sort name table entries by platformID – encodingID – languageID – nameID, or sort cmap table entires by Unicode codepoints.

Fixup (⌘+F) (CTRL+F)

Use this to remove duplicate table entries from tables of variable length (like name, cmap, kern). This is helpful, for example, if you have used **Grow** to duplicate an existing name table name record but then find that you do not need an additional name record.



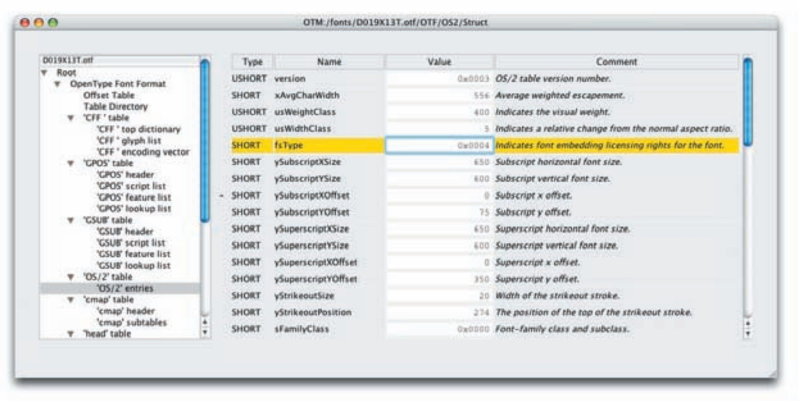
Viewing Tables

OTMaster offers two modes for viewing tables and two modes of table value representation representing table values.

VIEW MENU

Nested Tables (⌘+N) (CTRL+N)

This mode presents each table's content as a typical dialog. With tables like head, hhea, OS/2 there are, per entry, its data **Type**, its **Name** and finally the **Value** in a textbox which may be edited in most cases. Where possible, there is an additional **Comment** column with a brief description. This way, OTMaster is not just a table editor but at the same time provides a built-in documentation for most of the table data. For example, OTMaster exposes glyph names in the **Comment** column which is particularly useful since tables identify glyphs by mere glyph indices.



Type	Name	Value	Comment
USHORT	version	0x0003	*OS/2 table version number.*
SHORT	xAvgCharWidth	556	*Average weighted escapement.*
USHORT	usWeightClass	400	*Indicates the visual weight.*
USHORT	usWidthClass	5	*Indicates a relative change from the normal aspect ratio.*
SHORT	fsType	0x0004	*Indicates font embedding licensing rights for the font.*
SHORT	ySubscriptXSize	650	*Subscript horizontal font size.*
SHORT	ySubscriptYSize	600	*Subscript vertical font size.*
SHORT	ySubscriptXOffset	0	*Subscript x offset.*
SHORT	ySubscriptYOffset	75	*Subscript y offset.*
SHORT	ySuperscriptXSize	650	*Superscript horizontal font size.*
SHORT	ySuperscriptYSize	600	*Superscript vertical font size.*
SHORT	ySuperscriptXOffset	0	*Superscript x offset.*
SHORT	ySuperscriptYOffset	350	*Superscript y offset.*
SHORT	yStrikeoutSize	20	*Width of the strikeout stroke.*
SHORT	yStrikeoutPosition	274	*The position of the top of the strikeout stroke.*
SHORT	sFamilyClass	0x0000	*Font-family class and subclass.*

The Nested Tables mode. Edit table entries in usual textboxes!

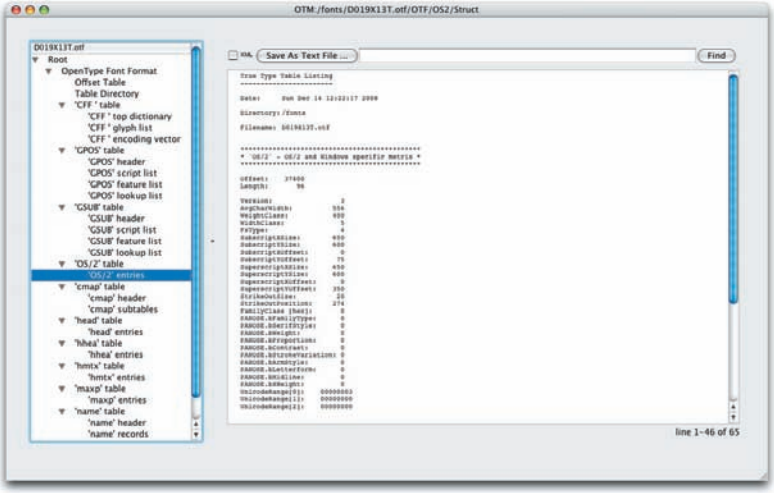
The Nested Tables mode exposes only the top level, indicating the existence of a nested table by a **+** sign which, when clicked, will show or hide a nested table's entries.

Double-click a (nested) table's header to display its content in a separate window.

While in Nested Tables mode, a right-click into the content area will open a context menu which offers **Text Dump** and **XML Dump** modes (see the next page for details). Choosing either of them will open a new window.

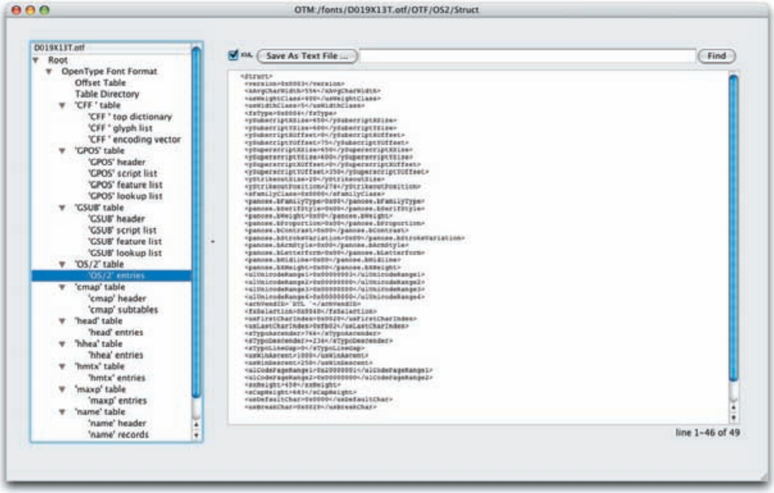
Text Dump (⌘+T) (CTRL+T)

Text Dump will present table data as text. Please note that with complex tables – like glyf, CFF, GSUB or GPOS – this text is rather a summary:



A table's content in Text Dump mode.

While in Text Dump mode, the **XML** checkbox at the top of the table content area allows you to switch to XML mode which will format the summary in XML style:

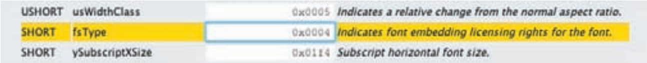


Text dump in XML style. Please note that this is not compatible with TTX's XML-style dump!

Save As Text File … will save a plain or XML text dump – useful for comparison in a text editor, or making printouts for proofreading. **Find** will search the (XML) text dump for a string.

hex (⌘+H) (CTRL+H)

Hex shows values as hexadecimal numbers:

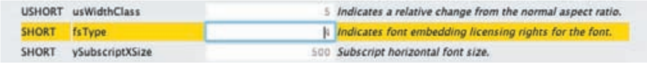


USHORT	usWidthClass	0x0005	*Indicates a relative change from the normal aspect ratio.*
SHORT	fsType	0x0004	*Indicates font embedding licensing rights for the font.*
SHORT	ySubscriptXSize	0x01F4	*Subscript horizontal font size.*

This example is taken from the OS/2 table entries.

dec (⌘+D) (CTRL+D)

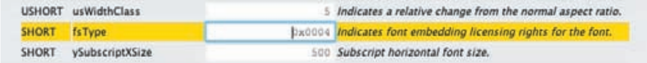
Dec shows values as decimal numbers:



USHORT	usWidthClass	5	*Indicates a relative change from the normal aspect ratio.*
SHORT	fsType	4	*Indicates font embedding licensing rights for the font.*
SHORT	ySubscriptXSize	500	*Subscript horizontal font size.*

best fitting (⌘+B) (CTRL+B)

Best fitting will choose the most appropriate representation of values, which is hexadecimal or decimal. This is the default.



USHORT	usWidthClass	5	*Indicates a relative change from the normal aspect ratio.*
SHORT	fsType	0x0004	*Indicates font embedding licensing rights for the font.*
SHORT	ySubscriptXSize	500	*Subscript horizontal font size.*

*You will notice that with **best fitting**, which is the default, some values are decimal while others are hexadecimal – this choice is built into OTMaster.*

OpenType Font Tables

You are expected to know the OpenType specification, rudimentarily at least, and at least as regards the tables whose data you intend to adjust. It is not the task of this manual to reproduce or rephrase them here, though this cannot be avoided at times. ▶ *local* ▶ *www* What you will find in this chapter though is a brief summary of an OpenType font's structure, a list of common tables, with a few additional notes about things to consider: first, to make sure that data across tables is consistent, and second, to provide some OTMaster tricks for adjusting and extending tables.

As indicated in the ▶ *Inspecting Tables* chapter, the structure of an OpenType font is quite simple. It starts with header information. First, an Offset Table whose SFNT version indicates whether the font is TT-based (0x001000), CFF-based (string 'OTTO') or a TrueType Collection (string 'ttcf'), and the number of tables in case of a TT/CFF-based OpenType font. Second, a Table Directory which points out where each table starts and how long it is. Header information are directly followed by the tables.

The OpenType specification provide the following list of tables:

1. Required tables

cmap	character to glyph mapping
head	font header
hhea	horizontal header
hmtx	horizontal metrics
maxp	maximum profile
name	naming table
OS/2	OS/2 and Windows specific metrics
post	PostScript information

2. Tables in TT-based OpenType fonts

cvt	control value table
fpgm	font program
glyf	glyph data
loca	index to location*
prep	CVT program

3. Tables in CFF-based OpenType fonts

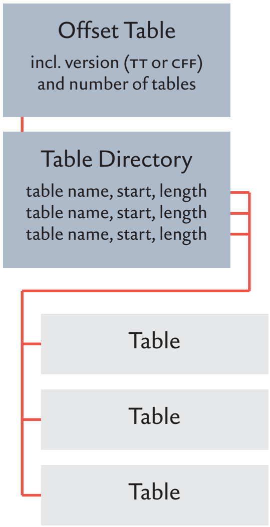
CFF	PostScript font program (Compact Font Format)†
VORG	vertical origin

4. Tables for bitmaps

EBDT	embedded bitmap data
EBLC	embedded bitmap location data
EBSC	embedded bitmap scaling data

Info: The OpenType specification. ▶ *local* ▶ *www* *Especially see* The OpenType Font File ▶ *local* ▶ *www and* Recommendations for OpenType Fonts ▶ *local* ▶ *www which comes with additional information and clarifications.*

An OpenType font's data structure (simplified):


Offset Table
incl. version (TT or CFF) and number of tables
Table Directory
table name, start, length
table name, start, length
table name, start, length
Table
Table
Table

---

* *Location means: location of glyphs' data inside of the* glyf *table. Effectively, the* loca *is the* glyf *table's 'external' header or directory.*

† *The* CFF *table actually is a font in its own right, holding not only outline data, but also glyph- and font-level hinting information, font names and more.*

5. Typographic layout tables

BASE	baseline data
GDEF	glyph definition data
GPOS	glyph positioning data
GSUB	glyph substitution data
JSTF	justification data

6. Other tables

DSIG	digital signature
gasp	grid-fitting and scan-conversion procedure
hdmx	horizontal device metrics
kern	kerning
LTSH	linear threshold data
PCLT	PCL5 data
VDMX	vertical device metrics
vhea	vertical metrics header
vmtx	vertical metrics

OpenType fonts' SFNT structure makes it easy to add tables which are not standardized by the OpenType specifications. The only requirement is that their names do not collide with standard table's names.

For example, Microsoft's VOLT – an application for defining typographic layout behavior visually – adds a table called TSIV to the font. This holds the VOLT project data. The TSIV data is removed from the font as soon as you 'ship' a font.

Selected Notes About OpenType Font Tables

Below, you will find some notes about individual OpenType font tables. These are in no way official recommendations but rather point out a few things to consider. You are strongly adviced to compare with the original OpenType specification – links are provided whereever possible.

Unsupported tables are listed in the table directory and table overview, but no content is shown in the table content area. When saving a font, these unsupported tables will be written back to the font in their original form.

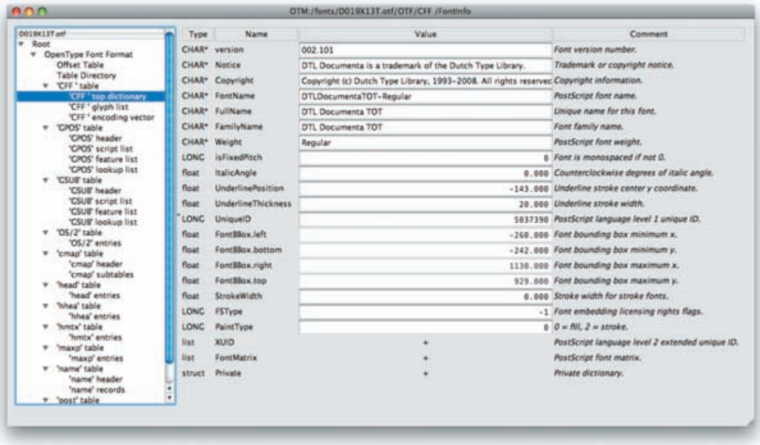
— AAT Tables

AAT (Apple Advanced Typography) tables are not supported.

— CFF

An OpenType font's Compact Font Format table is a complete font. It comes with outline data, but also with font names and (unless CID-keyed) glyph names, font- and glyph-wide hinting instructions, and more.

Info: The CFF *table.* ► *local* ► *www* *Adobe's* The Compact Font Format Specification, #5176. ► *local* ► *www* *Adobe's* The Type 2 Charstring Format #5177. ► *local* ► *www*



Type	Name	Value	Comment
CHAR*	version	002.101	*Font version number.*
CHAR*	Notice	DTL Documenta is a trademark of the Dutch Type Library.	*Trademark or copyright notice.*
CHAR*	Copyright	Copyright (c) Dutch Type Library, 1993-2008. All rights reservec	*Copyright information.*
CHAR*	FontName	DTLDocumentaTOT-Regular	*PostScript font name.*
CHAR*	FullName	DTL Documenta TOT	*Unique name for this font.*
CHAR*	FamilyName	DTL Documenta TOT	*Font family name.*
CHAR*	Weight	Regular	*PostScript font weight.*
LONG	isFixedPitch	0	*Font is monospaced if not 0.*
float	ItalicAngle	0.000	*Counterclockwise degrees of italic angle.*
float	UnderlinePosition	-143.000	*Underline stroke center y coordinate.*
float	UnderlineThickness	20.000	*Underline stroke width.*
LONG	UniqueID	5037390	*PostScript language level 1 unique ID.*
float	FontBBox.left	-260.000	*Font bounding box minimum x.*
float	FontBBox.bottom	-242.000	*Font bounding box minimum y.*
float	FontBBox.right	1130.000	*Font bounding box maximum x.*
float	FontBBox.top	929.000	*Font bounding box maximum y.*
float	StrokeWidth	0.000	*Stroke width for stroke fonts.*
LONG	FSType	-1	*Font embedding licensing rights flags.*
LONG	PaintType	0	*0 = fill, 2 = stroke.*
list	XUID	+	*PostScript language level 2 extended unique ID.*
list	FontMatrix	+	*PostScript font matrix.*
struct	Private	+	*Private dictionary.*

Note: Adobe discourages the use of XUID *and* UniqueID *in fonts that are not* CJK *fonts.* ► *www*

The 'CFF' top dictionary *which includes the* CFF *table's Name* INDEX *as* **FontName**.

1. **'CFF' top dictionary** contains a CFF font's meta-information:

1.1 CFF and name table: The CFF table's Name INDEX – found in the 'CFF' top dictionary as **FontName** – must be identical to name table entries with nameID 6 (*PostScript* name, both Microsoft platform and Macintosh platform) and nameID 4 (*Full* name, Microsoft platform). If your font is a CFF-based OpenType font and you intend to change these names, then you need to change the 'CFF' top dictionary's **FontName** at first and only then can adjust the name table entries for *PostScript* name and *Full* name.

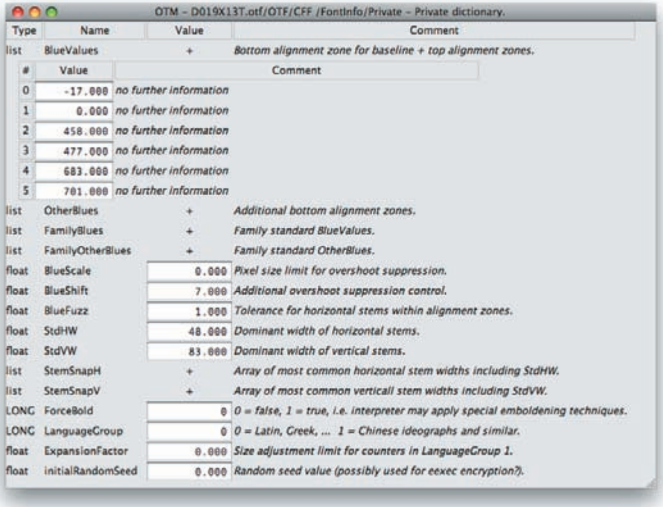
Other font name entries in the 'CFF' top dictionary may reflect the respective name table entries. **Weight** refers to weight only, not to style!

1.2 CFF and post table: Both the CFF and the post table have entries for underline position and thickness. CFF has **UnderlinePosition** and **UnderlineThickness**. post has **underlinePosition** and **underlineThickness**. While both tables' underline thickness entries share the same value, those for underline position differ. The CFF table's **UnderlinePosition** is measured from the vertical center of the underline outline's height, or thickness. The post table' **underlinePosition** is measured from its highest point. Their relation is:

CFF.UnderlinePosition
= post.underlinePosition – (post.underlineThickness / 2)

Take care that **ItalicAngle** is identical with the post table's **italicAngle** and matches the hhea table's **caretSlopeRise** and **caretSlopeRun**. More in the ► *Consistency Checker* chapter.

1.3 Font-wide hinting information – **BlueValues**, **OtherBlues**, etc. – are found in the 'CFF' top dictionary's **Private** nested table:



OTM – D019X13T.otf/OTF/CFF /FontInfo/Private – Private dictionary.

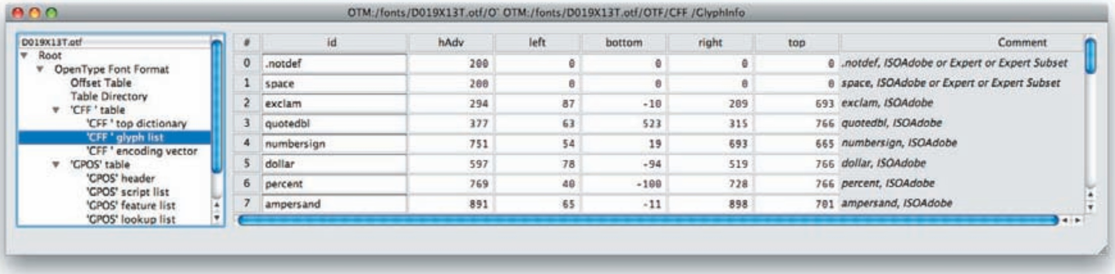
Type	Name	Value	Comment
list	BlueValues	+	*Bottom alignment zone for baseline + top alignment zones.*

#	Value	Comment
0	-17.000	*no further information*
1	0.000	*no further information*
2	458.000	*no further information*
3	477.000	*no further information*
4	683.000	*no further information*
5	701.000	*no further information*

Type	Name	Value	Comment
list	OtherBlues	+	*Additional bottom alignment zones.*
list	FamilyBlues	+	*Family standard BlueValues.*
list	FamilyOtherBlues	+	*Family standard OtherBlues.*
float	BlueScale	0.000	*Pixel size limit for overshoot suppression.*
float	BlueShift	7.000	*Additional overshoot suppression control.*
float	BlueFuzz	1.000	*Tolerance for horizontal stems within alignment zones.*
float	StdHW	48.000	*Dominant width of horizontal stems.*
float	StdVW	83.000	*Dominant width of vertical stems.*
list	StemSnapH	+	*Array of most common horizontal stem widths including StdHW.*
list	StemSnapV	+	*Array of most common verticall stem widths including StdVW.*
LONG	ForceBold	0	*0 = false, 1 = true, i.e. interpreter may apply special emboldening techniques.*
LONG	LanguageGroup	0	*0 = Latin, Greek, ... 1 = Chinese ideographs and similar.*
float	ExpansionFactor	0.000	*Size adjustment limit for counters in LanguageGroup 1.*
float	initialRandomSeed	0.000	*Random seed value (possibly used for eexec encryption?).*

*The **Private** dictionary which is accessed by way of the 'CFF' top dictionary – here displayed in a separate window.*

2. 'CFF' glyph list holds the glyph names as found in the CFF table's CharStrings INDEX. Additional columns present, per glyph,
hAdv, the horizontal advance width,
left, the leftside bearing which is equal to xMin (smallest horizontal extension of the glyph),
bottom, the yMin (smallest vertical extension of the glyph),
right, the xMax (largest horizontal extension of the glyph),
top, the yMax (largest vertical extension of the glyph),
and **Comment** with Adobe's standard glyph names for these glyphs.

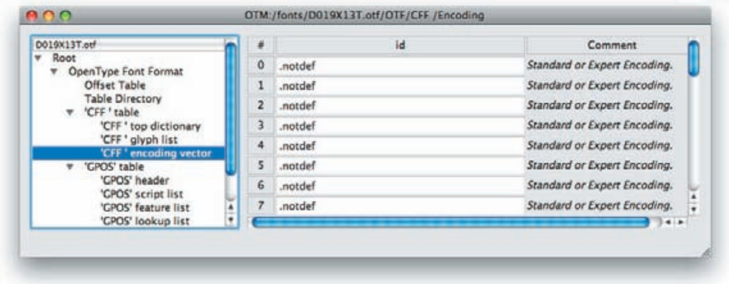


OTM:/fonts/D019X13T.otf/O' OTM:/fonts/D019X13T.otf/OTF/CFF /GlyphInfo

#	id	hAdv	left	bottom	right	top	Comment
0	.notdef	200	0	0	0	0	*.notdef, ISOAdobe or Expert or Expert Subset*
1	space	200	0	0	0	0	*space, ISOAdobe or Expert or Expert Subset*
2	exclam	294	87	-10	209	693	*exclam, ISOAdobe*
3	quotedbl	377	63	523	315	766	*quotedbl, ISOAdobe*
4	numbersign	751	54	19	693	665	*numbersign, ISOAdobe*
5	dollar	597	78	-94	519	766	*dollar, ISOAdobe*
6	percent	769	40	-100	728	766	*percent, ISOAdobe*
7	ampersand	891	65	-11	898	701	*ampersand, ISOAdobe*

The 'CFF' glyph list. This is where you may change glyph names in CFF-based OpenType fonts. in TT-based OpenType fonts, glyph names are found in the post table – unless this is a version 3 post table.

3. 'CFF' encoding vector links any of the characters covered in Adobe's Standard or Expert Encoding to glyph indices. This is read-only.



#	id	Comment
0	.notdef	*Standard or Expert Encoding.*
1	.notdef	*Standard or Expert Encoding.*
2	.notdef	*Standard or Expert Encoding.*
3	.notdef	*Standard or Expert Encoding.*
4	.notdef	*Standard or Expert Encoding.*
5	.notdef	*Standard or Expert Encoding.*
6	.notdef	*Standard or Expert Encoding.*
7	.notdef	*Standard or Expert Encoding.*

The 'CFF' encoding vector.

Unlike the glyf table, the CFF table does not show outline coordinates. For editing outlines you may prefer to use the ▶ *Glyph Editor*.

— cmap

The Character To Glyph Mapping table maps input Unicode codepoints to glyphs which are identified by glyph index (GID). It consists of one or more subtables. A standard OpenType fonts contain three of them (platformID – encodingID – format):

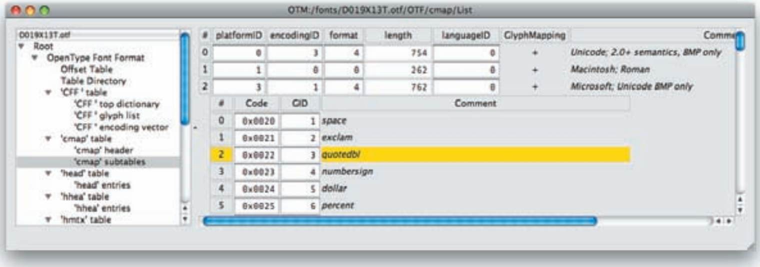
0 – 3 – 4 (Macintosh – Unicode – format 4)
1 – 0 – [6] (Macintosh – Roman – format 6 or other)
3 – 1 – 4 (Microsoft – Unicode – format 4)

The last one is required for Windows. The second one is or may become increasingly obsolete.

Info: The cmap *table.* ▶ *local* ▶ *www*

To add mapping information to a cmap subtable, click the **+** to access its entries, select one of them, choose **Edit > Grow** (⌘ + G) (CTRL + G) to duplicate the selected entry, and adjust Unicode codepoint and glyph index:

Note: You need to repeat this for every subtable if you want the according mappings to be covered by all of them.

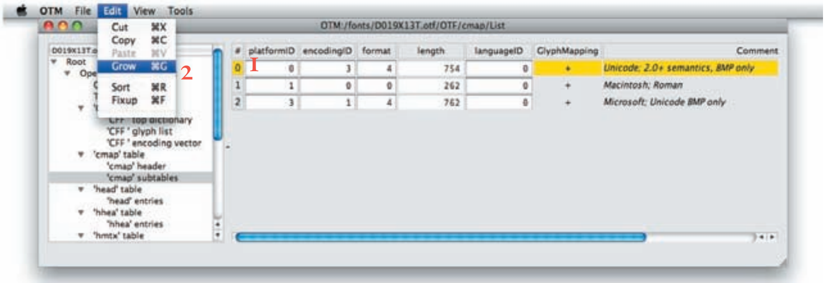


#	platformID	encodingID	format	length	languageID	GlyphMapping	Comme
0	0	3	4	754	0	+	Unicode; 2.0+ semantics, BMP only
1	1	0	0	262	0	+	Macintosh; Roman
2	3	1	4	762	0	+	Microsoft; Unicode BMP only

#	Code	GID	Comment
0	0x0020	1	space
1	0x0021	2	exclam
2	0x0022	3	quotedbl
3	0x0023	4	numbersign
4	0x0024	5	dollar
5	0x0025	6	percent

cmap *table with three subtables, the last of which is unfolded.*

It is possible to create a new subtables of a different format by
1. selecting an existing subtable,
2. duplicating it with **Grow**,



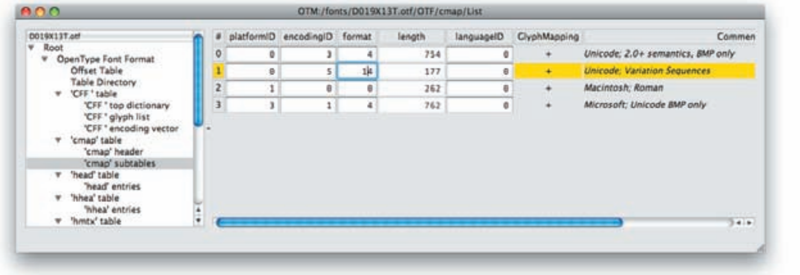
#	platformID	encodingID	format	length	languageID	GlyphMapping	Comment
0	0	3	4	754	0	+	Unicode; 2.0+ semantics, BMP only
1	1	0	0	262	0	+	Macintosh; Roman
2	3	1	4	762	0	+	Microsoft; Unicode BMP only

Duplicating a cmap *subtable.*

and finally changing the new subtable's format and possibly platformID and encodingID too.

OTMaster supports Unicode Variation Sequences. With the method described above it is possible to add an according subtable. For this, platformID – encodingID – format need to be 0 – 5 – 14.

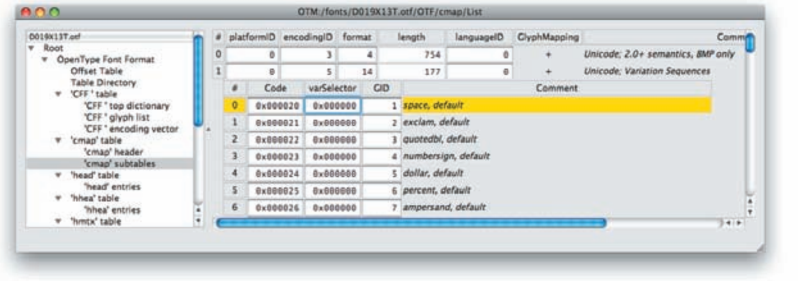
Note: See the cmap *table specification, 'Format 14: Unicode Variation Sequence'.* ►*local* ►*www*



#	platformID	encodingID	format	length	languageID	GlyphMapping	Commen
0	0	3	4	754	0	+	Unicode; 2.0+ semantics, BMP only
1	0	5	14	177	0	+	Unicode; Variation Sequences
2	1	0	0	262	0	+	Macintosh; Roman
3	3	1	4	762	0	+	Microsoft; Unicode BMP only

Creating a subtable for Unicode Variation Sequences. Change the duplicate subtable's ***format****,* ***platformID*** *and* ***encodingID****.*

A Variation Sequence is a codepoint (**Code**) followed by a variation selector (**varSelector**). If this pair is matched in an input string, the glyph mapped to this codepoint will be replaced by another glyph referenced by **GID**. This can be compared to a typographic layout feature, yet it is the input string itself which initiates the replacement by a variant glyph. By default, glyphs' variation selectors are set to zero and need to be adjusted:



#	platformID	encodingID	format	length	languageID	GlyphMapping	Comm
0	0	3	4	754	0	+	Unicode; 2.0+ semantics, BMP only
1	0	5	14	177	0	+	Unicode; Variation Sequences

#	Code	varSelector	GID	Comment
0	0x000020	0x000000	1	space, default
1	0x000021	0x000000	2	exclam, default
2	0x000022	0x000000	3	quotedbl, default
3	0x000023	0x000000	4	numbersign, default
4	0x000024	0x000000	5	dollar, default
5	0x000025	0x000000	6	percent, default
6	0x000026	0x000000	7	ampersand, default

*Adjust the glyph's variation selector (****varSelector****).*

Tip: Use ↑ *(arrow up) and* ↓ *(arrow down) keys to flip through all mapping entries quickly, and* ⇥ *(to right) and* ⇧+⇥ *(or* ⇤*; to left) to jump from column to column.*

— DSIG

If the Digital Signature table exists in a font which you are about to edit in OTMaster you may want to delete it with **Edit > Cut** (⌘+x) (CTRL+X). Editing the font would invalidate this table anyway and indicate that this font is not in the state in which it originally delivered.

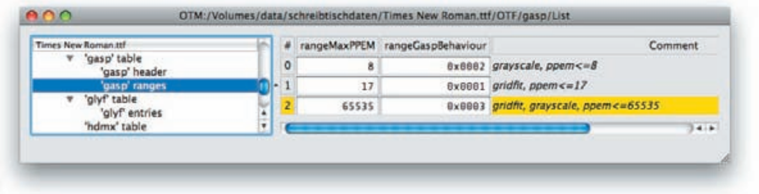
Info: The DSIG table. ►local ►www

Note: It is assumed that you are either editing your own font or have permission to do so!

— gasp

The Grid-Fitting And Scan-Conversion Procedure table determines which kind of rasterization is preferred for different ppem size ranges.

Info: The gasp table. ►local ►www



#	rangeMaxPPEM	rangeGaspBehaviour	Comment
0	8	0x0002	grayscale, ppem<=8
1	17	0x0001	gridfit, ppem<=17
2	65535	0x0003	gridfit, grayscale, ppem<=65535

Selecting the last record of the gasp table.

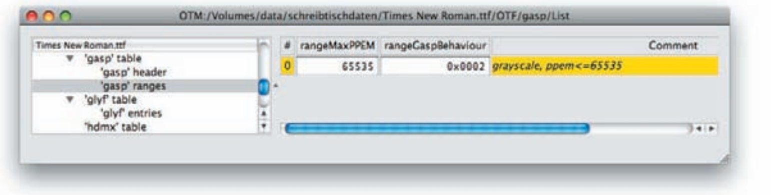
The upper limit of a range is its maximum ppem size (**rangeGaspPPEM**), the lower limit is 0 or the previous range's maximum ppem size +1. The last record must have a **rangeGaspPPEM** of 0xFFFF or 65 535. Four kinds of preferred rasterization (**rangeGaspBehavior**) have been defined:

grayscale (0x0002)
gridfit (0x0001)
grayscale and gridfit (0x0003)
none of them (0x0000)

There are three additional values which address ClearType:

symmetric gritfit (0x0004)
symmetric smoothing (0x0008)
symmetric gritfit and symmetric smoothing (0x000C)

For example, to adjust the above gasp table such that it will encourage grayscaling for all ppem sizes, select the first two records one by one and **Edit > Cut** (⌘+x) (CTRL+X) them. Finally, change **rangeGaspBehavior** to '**0x0002**'.



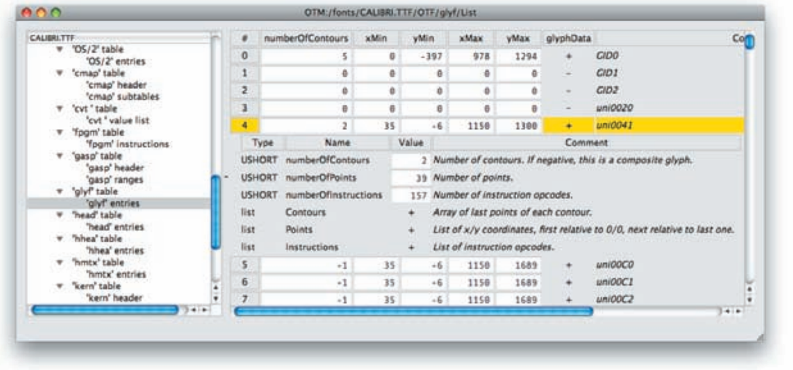
#	rangeMaxPPEM	rangeGaspBehaviour	Comment
0	65535	0x0002	grayscale, ppem<=65535

*The gasp table after removing the first two records and adjusting **rangeGaspBehavior**.*

— glyf
— loca

The glyf table contains TT outline information as well as instructions ('hints'). This table is read-only in OTMaster, so we do not go into details here. Try the ▸ *Glyph Editor* for editing glyph outlines.

Info: The glyf *table.* ▸ *local* ▸ *www*
The loca *table.* ▸ *local* ▸ *www*



#	numberOfContours	xMin	yMin	xMax	yMax	glyphData		Co
0	5	0	-397	978	1294	+	GID0	
1	0	0	0	0	0	-	GID1	
2	0	0	0	0	0	-	GID2	
3	0	0	0	0	0	-	uni0020	
4	2	35	-6	1150	1300	+	uni0041	

Type	Name	Value	Comment
USHORT	numberOfContours	2	Number of contours. If negative, this is a composite glyph.
USHORT	numberOfPoints	39	Number of points.
USHORT	numberOfInstructions	157	Number of instruction opcodes.
list	Contours	+	Array of last points of each contour.
list	Points	+	List of x/y coordinates, first relative to 0/0, next relative to last one.
list	Instructions	+	List of instruction opcodes.

#	numberOfContours	xMin	yMin	xMax	yMax	glyphData	
5	-1	35	-6	1150	1689	+	uni00C0
6	-1	35	-6	1150	1689	+	uni00C1
7	-1	35	-6	1150	1689	+	uni00C2

The loca table (not shown here) can be considered as the glyf table's directory or header. Per every glyph index, the loca table points out (by way of an 'offset') where this glyph's data starts in the glyf table.

Just like tables' header information are read-only, the loca table is read-only and will be updated automatically by OTMaster.

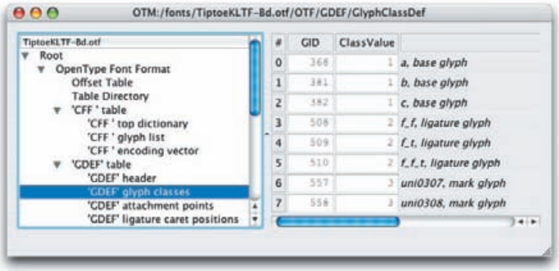
— GDEF

The Glyph Definition table belongs to the typographic layout tables and provides additional information to which GSUB and GDEF may refer.

Info: The GDEF *table.* ▶ *local* ▶ *www Also see the* OpenType Layout Common Table Formats document. ▶ *local* ▶ *www*

Curently, OTMaster only supports glyph class definitions which are located in 'GDEF' glyph classes. Each glyph, identified by glyph index (GID), is associated with no or one of these classes, identified by **ClassValue**:

1. base glyphs
2. ligature glyphs
3. (combining) mark glyphs
4. glyph components



OTM:/fonts/TiptoeKLTF-Bd.otf/OTF/GDEF/GlyphClassDef

TiptoeKLTF-Bd.otf
- Root
  - OpenType Font Format
    - Offset Table
    - Table Directory
    - 'CFF ' table
      - 'CFF ' top dictionary
      - 'CFF ' glyph list
      - 'CFF ' encoding vector
    - 'GDEF' table
      - 'GDEF' header
      - 'GDEF' glyph classes
      - 'GDEF' attachment points
      - 'GDEF' ligature caret positions

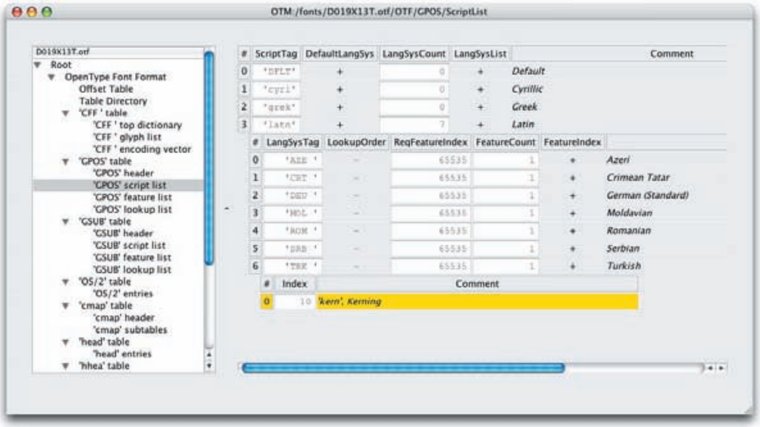
#	GID	ClassValue	
0	368	1	*a, base glyph*
1	381	1	*b, base glyph*
2	382	1	*c, base glyph*
3	508	2	*f_f, ligature glyph*
4	509	2	*f_t, ligature glyph*
5	510	2	*f_f_t, ligature glyph*
6	557	3	*uni0307, mark glyph*
7	558	3	*uni0308, mark glyph*

— **GPOS**

— **GSUB**

Glyph Positioning and the Glyph Substitution tables start with three lists: script list, language list, and feature list.

Info: The GPOS *table.* ▶ *local* ▶ *www The* GSUB *table.* ▶ *local* ▶ *www Also see the* OpenType Layout Common Table Formats document. ▶ *local* ▶ *www*



#	ScriptTag	DefaultLangSys	LangSysCount	LangSysList	Comment
0	'DFLT'	+	0	+	*Default*
1	'cyrl'	+	0	+	*Cyrillic*
2	'grek'	+	0	+	*Greek*
3	'latn'	+	7	+	*Latin*

#	LangSysTag	LookupOrder	ReqFeatureIndex	FeatureCount	FeatureIndex	
0	'AZE '	-	65535	1	+	*Azeri*
1	'CRT '	-	65535	1	+	*Crimean Tatar*
2	'DEU '	-	65535	1	+	*German (Standard)*
3	'MOL '	-	65535	1	+	*Moldavian*
4	'ROM '	-	65535	1	+	*Romanian*
5	'SRB '	-	65535	1	+	*Serbian*
6	'TRK '	-	65535	1	+	*Turkish*

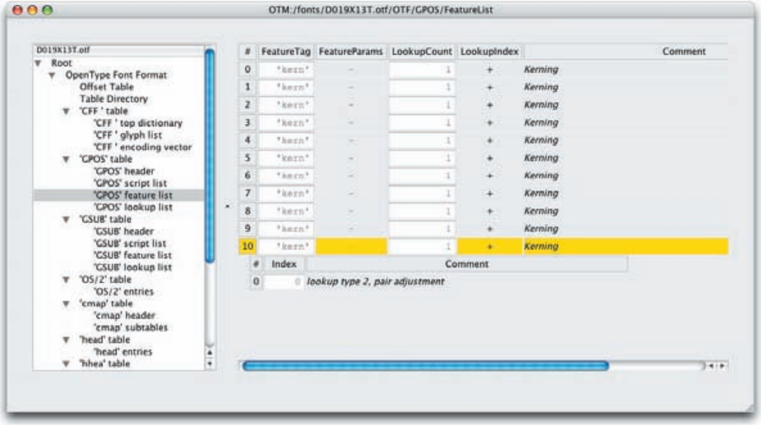
#	Index	Comment
0	10	*'kern', Kerning*

The script list, with unfolded language list for the 'latn' script.

The script list sums up all scripts explicitly addressed by the layout table. These are scripts – or writing systems – like 'latn' for Latin or 'cyrl' for Cyrillic.

Per each script as found in the **ScriptTag** column, there is a list of languages explicitly addressed by the layout table, plus a 'default' language for which there is a special place in the data structure. (In case that a layout application cannot find a match for the selected language – e.g. by way of the spelling dictionary in Adobe InDesign – in the font's layout table, it would fall back to this 'default' language.) The former is accessed by clicking the **+** in the **LangSysList** column, the latter by clicking the **+** in the **DefaultLangSys** column. In the example above, the 'latn' script's languages are folded out.

Per each language there is a list of features associated with it. This is unfolded by clicking the **+** in the according language's **FeatureIndex** column. In the example above, there is one 'kern' feature associated with the Turkish language or 'TRK' – to know what the actual feature behavior of 'kern' is for Turkish, we momorize feature **Index** 10 and switch to the feature list.



OTM:/fonts/D019X13T.otf/OTF/GPOS/FeatureList

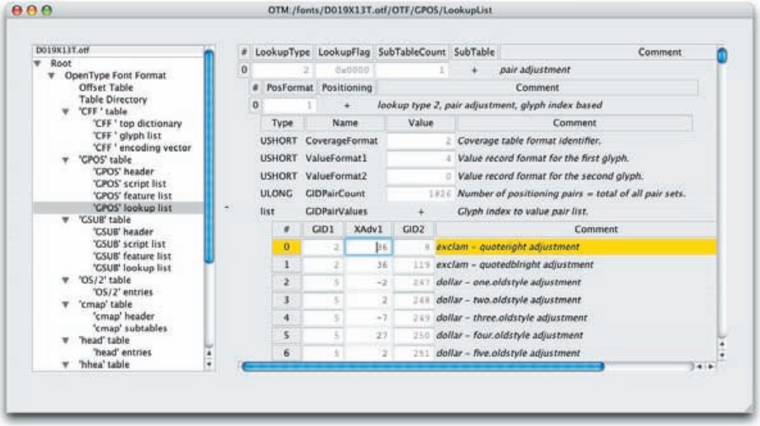
D019X13T.otf
- Root
  - OpenType Font Format
    - Offset Table
    - Table Directory
    - 'CFF ' table
      - 'CFF ' top dictionary
      - 'CFF ' glyph list
      - 'CFF ' encoding vector
    - 'GPOS' table
      - 'GPOS' header
      - 'GPOS' script list
      - 'GPOS' feature list
      - 'GPOS' lookup list
    - 'GSUB' table
      - 'GSUB' header
      - 'GSUB' script list
      - 'GSUB' feature list
      - 'GSUB' lookup list
    - 'OS/2' table
      - 'OS/2' entries
    - 'cmap' table
      - 'cmap' header
      - 'cmap' subtables
    - 'head' table
      - 'head' entries
    - 'hhea' table

#	FeatureTag	FeatureParams	LookupCount	LookupIndex	Comment
0	'kern'	–	1	+	*Kerning*
1	'kern'	–	1	+	*Kerning*
2	'kern'	–	1	+	*Kerning*
3	'kern'	–	1	+	*Kerning*
4	'kern'	–	1	+	*Kerning*
5	'kern'	–	1	+	*Kerning*
6	'kern'	–	1	+	*Kerning*
7	'kern'	–	1	+	*Kerning*
8	'kern'	–	1	+	*Kerning*
9	'kern'	–	1	+	*Kerning*
10	'kern'	–	1	+	*Kerning*

#	Index	Comment
0	0	*lookup type 2, pair adjustment*

The Feature list.

The feature list, for each feature to which a script/language combination refers, points to one or more lookups via **LookupIndex**. These lookups define the actual substitution and positioning behavior for this feature. Going back to our example, we remember feature index number 10 (the **#** column) and in this feature's **LookupIndex** column click the **+** which opens a nested table pointing to the relevant lookups' indices.



OTM:/fonts/D019X13T.otf/OTF/GPOS/LookupList

D019X13T.otf
- Root
  - OpenType Font Format
    - Offset Table
    - Table Directory
    - 'CFF ' table
      - 'CFF ' top dictionary
      - 'CFF ' glyph list
      - 'CFF ' encoding vector
    - 'GPOS' table
      - 'GPOS' header
      - 'GPOS' script list
      - 'GPOS' feature list
      - 'GPOS' lookup list
    - 'GSUB' table
      - 'GSUB' header
      - 'GSUB' script list
      - 'GSUB' feature list
      - 'GSUB' lookup list
    - 'OS/2' table
      - 'OS/2' entries
    - 'cmap' table
      - 'cmap' header
      - 'cmap' subtables
    - 'head' table
      - 'head' entries
    - 'hhea' table

#	LookupType	LookupFlag	SubTableCount	SubTable	Comment
0	2	0x0000	1	+	*pair adjustment*

#	PosFormat	Positioning	Comment
0	1	+	*lookup type 2, pair adjustment, glyph index based*

Type	Name	Value	Comment
USHORT	CoverageFormat	2	*Coverage table format identifier.*
USHORT	ValueFormat1	4	*Value record format for the first glyph.*
USHORT	ValueFormat2	0	*Value record format for the second glyph.*
ULONG	GIDPairCount	1826	*Number of positioning pairs = total of all pair sets.*
list	GIDPairValues	+	*Glyph index to value pair list.*

#	GID1	XAdv1	GID2	Comment
0	2	36	8	*exclam – quoteright adjustment*
1	2	36	119	*exclam – quotedblright adjustment*
2	5	-2	247	*dollar – one.oldstyle adjustment*
3	5	2	248	*dollar – two.oldstyle adjustment*
4	5	-7	249	*dollar – three.oldstyle adjustment*
5	5	27	250	*dollar – four.oldstyle adjustment*
6	5	2	251	*dollar – five.oldstyle adjustment*

The Lookup list.

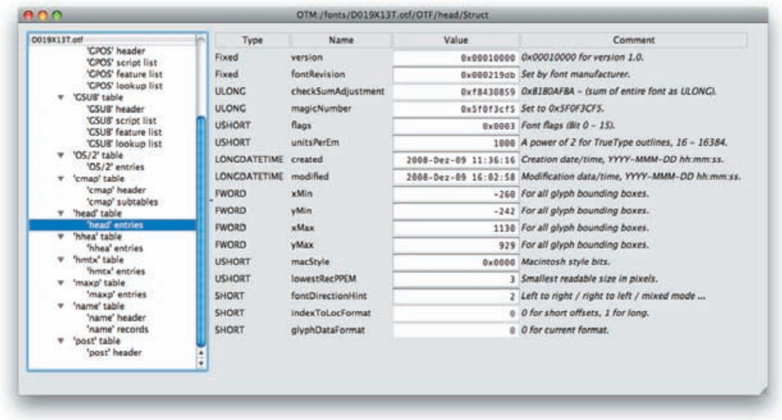
The lookup list, finally, holds a list of all lookups. These are presented as nested tables, and you can start unfolding nested tables which strictly represent the lookups' structure. In our example above, there is but a single lookup for kerning, of **LookupType** 2. We click the **+** in the **SubTable** column and get to know about the single subtable's format which is **PosFormat** 1. Clicking the **+** in the **Positioning** column unfolds the single subtable's header information. Clicking the **+** in the **GIDPairValues** row finally presents the kerning pairs and values.

Note: The structure of nested tables depends on lookup types and subtable formats – this example really is just that: an example.

— head

The Font Header table holds the most important value – the font's **unitsPerEm** or upm, i.e. the number of units into which the em-square is divided. All of the fonts metrics relate to it.

Info: The head *table.* ▶ *local* ▶ *www*



OTM:/fonts/D019X13T.otf/OTF/head/Struct

Type	Name	Value	Comment
Fixed	version	0x00010000	0x00010000 for version 1.0.
Fixed	fontRevision	0x000219db	Set by font manufacturer.
ULONG	checkSumAdjustment	0xf8430859	0xB180AFBA – (sum of entire font as ULONG).
ULONG	magicNumber	0x5f0f3cf5	Set to 0x5F0F3CF5.
USHORT	flags	0x0003	Font flags (Bit 0 – 15).
USHORT	unitsPerEm	1000	A power of 2 for TrueType outlines, 16 – 16384.
LONGDATETIME	created	2008-Dez-09 11:36:16	Creation date/time, YYYY-MMM-DD hh:mm:ss.
LONGDATETIME	modified	2008-Dez-09 16:02:58	Modification data/time, YYYY-MMM-DD hh:mm:ss.
FWORD	xMin	-260	For all glyph bounding boxes.
FWORD	yMin	-242	For all glyph bounding boxes.
FWORD	xMax	1130	For all glyph bounding boxes.
FWORD	yMax	929	For all glyph bounding boxes.
USHORT	macStyle	0x0000	Macintosh style bits.
USHORT	lowestRecPPEM	3	Smallest readable size in pixels.
SHORT	fontDirectionHint	2	Left to right / right to left / mixed mode ...
SHORT	indexToLocFormat	0	0 for short offsets, 1 for long.
SHORT	glyphDataFormat	0	0 for current format.

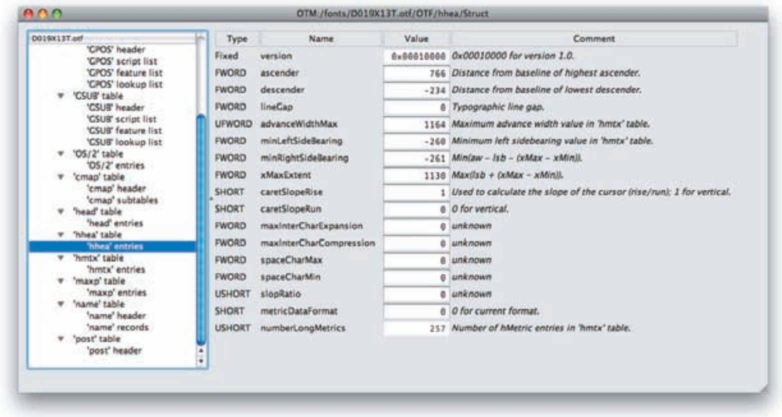
head *table entries.*

macStyle relates to the OS/2 table's **fsSelection** (and **usWeightClass** and **usWidthClass**) and name table records with nameID 2.

— hhea

Many Horizontal Header table entries are not meant to be edited manually.

Info: The hhea *table.* ▶ *local* ▶ *www*



OTM:/fonts/D019X13T.otf/OTF/hhea/Struct

Type	Name	Value	Comment
Fixed	version	0x00010000	0x00010000 for version 1.0.
FWORD	ascender	766	Distance from baseline of highest ascender.
FWORD	descender	-234	Distance from baseline of lowest descender.
FWORD	lineGap	0	Typographic line gap.
UFWORD	advanceWidthMax	1164	Maximum advance width value in 'hmtx' table.
FWORD	minLeftSideBearing	-260	Minimum left sidebearing value in 'hmtx' table.
FWORD	minRightSideBearing	-261	Min(aw – lsb – (xMax – xMin)).
FWORD	xMaxExtent	1130	Max(lsb + (xMax – xMin)).
SHORT	caretSlopeRise	1	Used to calculate the slope of the cursor (rise/run); 1 for vertical.
SHORT	caretSlopeRun	0	0 for vertical.
FWORD	maxInterCharExpansion	0	unknown
FWORD	maxInterCharCompression	0	unknown
FWORD	spaceCharMax	0	unknown
FWORD	spaceCharMin	0	unknown
USHORT	slopRatio	0	unknown
SHORT	metricDataFormat	0	0 for current format.
USHORT	numberLongMetrics	257	Number of hMetric entries in 'hmtx' table.

hhea *table entries.*

caretSlopeRise and **caretSlopeRun** should be in tune with the post table's **italicAngle** value and, if this is a CFF-based OpenType font, also with the 'CFF' top dictionary's **ItalicAngle**.

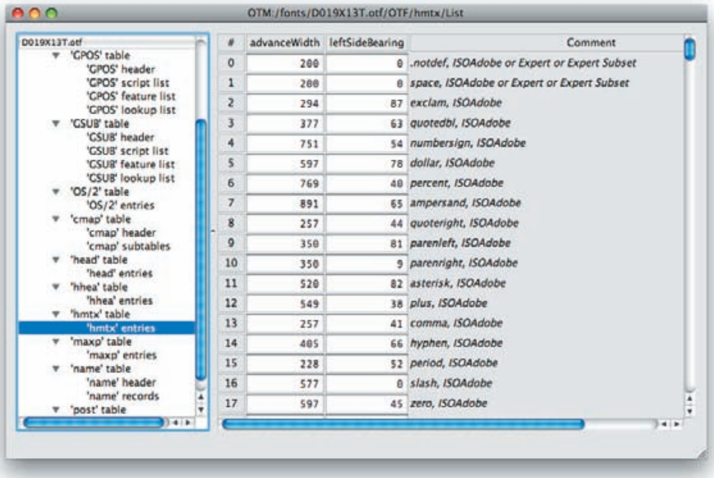
Note: For both italic/slant information and vertical font metrics see the ▶ Consistency Checker *chapter.*

ascender, **descender**, **lineGap** relate to OS/2 table's **sTypoAscender**, **sTypoDescender**, **sTypoLineGap**, **usWinAscent**, **usWinDescent**.

— hmtx

The Horizontal Metrics table contains, per glyph, **advanceWidth** and **leftSideBearing**.

Info: The hmtx *table.* ▶ *local* ▶ *www*



#	advanceWidth	leftSideBearing	Comment
0	200	0	.notdef, ISOAdobe or Expert or Expert Subset
1	200	0	space, ISOAdobe or Expert or Expert Subset
2	294	87	exclam, ISOAdobe
3	377	63	quotedbl, ISOAdobe
4	751	54	numbersign, ISOAdobe
5	597	78	dollar, ISOAdobe
6	769	40	percent, ISOAdobe
7	891	65	ampersand, ISOAdobe
8	257	44	quoteright, ISOAdobe
9	350	81	parenleft, ISOAdobe
10	350	9	parenright, ISOAdobe
11	520	82	asterisk, ISOAdobe
12	549	38	plus, ISOAdobe
13	257	41	comma, ISOAdobe
14	405	66	hyphen, ISOAdobe
15	228	52	period, ISOAdobe
16	577	0	slash, ISOAdobe
17	597	45	zero, ISOAdobe

The basic glyph metrics, advance width and left sidebearing, listed in the hmtx *table.*

In CFF-based OpenType fonts, hmtx table entries are read-only and cannot be edited.

— kern

Info: The kern *table.* ▶ *local* ▶ *www*

Theoretically speaking, the Kerning table is a remainder of the (pre-OpenType) TrueType format. Practically speaking, it is still required because many applications cannot read OpenType font's layout tables.

With OpenType fonts, kerning information better be provided by way of a kern feature in the GPOS table. This can deal with 'normal' left-to-right as well as right-to-left and vertical kerning. The specification even says – albeit strangely worded – that CFF-based OpenType fonts are not supposed to have a kern table at all.

Since some applications, even most popular ones, do not support typographic layout features (GSUB, GPOS etc.), unfortunately there is no way around adding a kern table to provide them with kerning information.

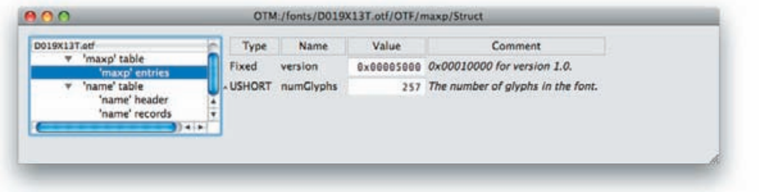
However, adding a kern table brings some inconveniences with it. Fonts are getting bigger and in turn require more kerning pairs. For a kern table to be recognized by Windows, it must contain a single subtable of format 0. Since the format 0 subtable's value which indicates the number of kerning pairs can be 65 535 at maximum (not to mention that the value which indicates the subtable's length is similarly restricted),* the number of kerning pairs that may go into such a subtable is limited – not enough to cover all pairs which would result from expanding class-based kerning as found in an OpenType font's GPOS 'kern' feature.

More about kern table editing in the ▶ *'kern' Table Viewer* chapter.

— maxp

Info: The maxp *table.* ▶ *local* ▶ *www*

The small Maximum Profile table tells about the number of glyphs (**numGlyphs**) in a font.



Type	Name	Value	Comment
Fixed	version	0x00005000	*0x00010000 for version 1.0.*
USHORT	numGlyphs	257	*The number of glyphs in the font.*

The maxp *table and its two entries.*

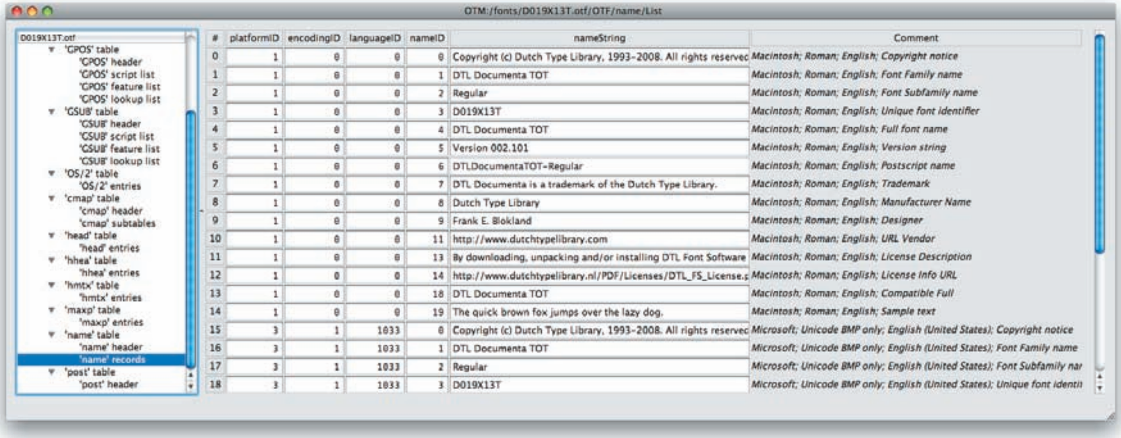
---

* *A detailed description was posted to the OpenType List in 2008 by Ascender Corp.'s Joshua Hadley.*

— name

The Naming table is essential for a font to be identified.

Info: The name *table.* ▶ *local* ▶ *www*



#	platformID	encodingID	languageID	nameID	nameString	Comment
0	1	0	0	0	Copyright (c) Dutch Type Library, 1993-2008. All rights reservec	*Macintosh; Roman; English; Copyright notice*
1	1	0	0	1	DTL Documenta TOT	*Macintosh; Roman; English; Font Family name*
2	1	0	0	2	Regular	*Macintosh; Roman; English; Font Subfamily name*
3	1	0	0	3	D019X13T	*Macintosh; Roman; English; Unique font identifier*
4	1	0	0	4	DTL Documenta TOT	*Macintosh; Roman; English; Full font name*
5	1	0	0	5	Version 002.101	*Macintosh; Roman; English; Version string*
6	1	0	0	6	DTLDocumentaTOT-Regular	*Macintosh; Roman; English; Postscript name*
7	1	0	0	7	DTL Documenta is a trademark of the Dutch Type Library.	*Macintosh; Roman; English; Trademark*
8	1	0	0	8	Dutch Type Library	*Macintosh; Roman; English; Manufacturer Name*
9	1	0	0	9	Frank E. Blokland	*Macintosh; Roman; English; Designer*
10	1	0	0	11	http://www.dutchtypelibrary.com	*Macintosh; Roman; English; URL Vendor*
11	1	0	0	13	By downloading, unpacking and/or installing DTL Font Software	*Macintosh; Roman; English; License Description*
12	1	0	0	14	http://www.dutchtypelibrary.nl/PDF/Licenses/DTL_FS_License.p	*Macintosh; Roman; English; License Info URL*
13	1	0	0	18	DTL Documenta TOT	*Macintosh; Roman; English; Compatible Full*
14	1	0	0	19	The quick brown fox jumps over the lazy dog.	*Macintosh; Roman; English; Sample text*
15	3	1	1033	0	Copyright (c) Dutch Type Library, 1993-2008. All rights reservec	*Microsoft; Unicode BMP only; English (United States); Copyright notice*
16	3	1	1033	1	DTL Documenta TOT	*Microsoft; Unicode BMP only; English (United States); Font Family name*
17	3	1	1033	2	Regular	*Microsoft; Unicode BMP only; English (United States); Font Subfamily nar*
18	3	1	1033	3	D019X13T	*Microsoft; Unicode BMP only; English (United States); Unique font identii*

The name *table entries. Thanks to the* **Comment** *column, the numeric platformID, encodingID, languageID and nameID are self-explaining.*

You may remove, edit or add new name entries. To add a name entry, select an existing one and click **Edit > Grow** (⌘+G) (CTRL+G) to duplicate it. Adjust the platformID, encodingID, languageID, nameID and the name tag. Once you have finished adding name entries, click **Edit > Sort** (⌘+R) (CTRL+R) which will reorder all name entries.

Pre-defined nameIDs are, according to the name table specification:

0. *Copyright* notice.
1. *Family* name. Each family, i.e. a collection of fonts which share the same *Family* name, is supposed to consist of no more than these four styles: 'Regular', 'Italic', 'Bold', 'Bold Italic', defined in the *Subfamily* name (2).
2. *Subfamily* (or *Style*) name. For each family – defined by *Family* name (1) – this may be one of 'Regular', 'Italic', 'Bold', 'Bold Italic'. The OS/2 table's **fsSelection** ▶ *local* ▶ *www* and the head table's **macStyle** ▶ *local* ▶ *www* need to reflect the style defined by *Subfamily* name.
3. *Unique Font Identifier*. This could be a combination of *Designer* (9) or *Manufacturer* (8) name, *Postscript* name (6) and year, separated by ';'.
4. *Full* name. A combination of *Preferred Family* name (16) and *Preferred Subfamily* name (17) if present or – according to the specification – *Family* name (1) and *Subfamily* name (2). If the Microsoft platform *Subfamily* name (2) is 'Regular', the *Full* name should match this platform's *Family* name (1).
5. *Version* string. This should begin with 'Version [major].[minor]' whereby [major] and [minor] may be any number smaller than 65 535. A *Version* string could look like 'Version 1.500'. Additional information may be added and should be separated by ';'.

Tip: For details see the name *table specification as well as the* ▶ Consistency Checker. *Additional information can be found in the* MakeOTF User Guide *which is part of the* AFDKO *download, especially the chapter* New OS/2 Bits. ▶ *local* ▶ *www Additional* name *table entries and the version 4* OS/2 *table's* **fsSelection** *bits 7–9 relate to* WPF*'s font selection model. See Microsoft's paper* WPF Font Selection Model ▶ *www for detailed information about WWS-conformant families.*

Note: The PostScript *name (6) description mentions an additional condition the* Full *name (4) needs to meet in* CFF*-based OpenType fonts.*

6. *PostScript* name. This is required for a PostScript interpreter to identify a PostScript language font corresponding to this OpenType font. There must be a record with nameID 6 for both (platformID–encodingID–languageID):

1–0–0	(Macintosh)
3–1–1033	(Microsoft)

Both strings must be identical when translated to ASCII and not longer than 63 characters. Characters are restricted to ASCII codes 33–126 excluding any of '**[](){}<>/%**'. Usually built like the *Full* name (4) – but without any spaces and with with a '-' between family and subfamily part of the string.

There is an additional requirement for CFF-based OpenType fonts: The *PostScript* name (6) for both Microsoft platform and Macintosh platform, the *Full* name (4) for the Microsoft platform as well as the CFF Name INDEX must be identical. OTMaster displays the CFF Name INDEX as **FontName** inside of the 'CFF' top dictionary, thereby deviating a bit from the CFF table's structure.

7. *Trademark* notice.

8. *Manufacturer* name.

9. *Designer* name.

10. *Description* of the typeface.

11. *Vendor* URL. This should include 'http://', 'ftp://' etc.

12. *Designer* URL. This should include 'http://', 'ftp://' etc.

13. *License* description. This is expected to be a summary of the terms rather than, as the specification puts it, 'legalese'.

14. *License Info* URL. A link to the full license text. This should include 'http://', 'ftp://' etc.

15. Reserved, set to zero.

16. *Preferred Family* name. If a family consists of more, or other, styles than 'Regular', 'Italic', 'Bold', 'Bold Italic', you may define a *Preferred Family* name for which there are no restrictions as to the number of styles.

17. *Preferred Subfamily* (or *Style*) name. This relates to the *Preferred Family* name (16). If the OS/2 table's version is 4 and if a family's *Preferred* styles are wws-conformant (i.e. are distinguished by the categories weight, width and slope alone), you may set the OS/2 table's **fsSelection** bit 8 to 1 which will signal wws-conformancy. If a family's *Preferred* styles are not wws-conformant, set bit 8 to 0 and provide wws-conformant *wws Family* name (21) and *wws Subfamily* name (22).

Note: If you set the OS/2 *table's version to 4, then all the* ***fsSelection*** *bits 7–9 need to be set to 0 or 1 consciously because starting with this table version, these bit settings bear a special meaning.*

18. *Compatible Full* name. Macintosh platform only. If the *Full* name (4) turns out to be too long, i.e. longer than 27 (or more generously: 31) characters, you may either abbreviate the Macintosh platform *Full* name (4) or otherwise provide an additional abbreviated *Compatible Full* name. If the *Compatible Full* name is present and the *Subfamily* name (2) is 'Regular', the *Compatible Full* name should match the *Family* name (1).

19. *Sample* text.

20. *PostScript* CID findfont name. Please see the name table specification.

21. *wws Family* name. A wws-conformant family's styles must address no other categories than weight, width and slope. If this name is provided, the OS/2 table's **fsSelection** bit 8 should be set to 0.
22. *wws Subfamily* (or *Style*) name. If this name is provided, the OS/2 table's **fsSelection** bit 8 should be set to 0. The OS/2 table's **usWeightClass**, **usWidthClass** as well as **fsSelection** bits 0 (italic) and 9 (oblique) should reflect the *wws Subfamily* name.

The name table specification ▶ *local* ▶ *www* comes with an example string for each nameID, so that you are encouraged to read this documents carefully.

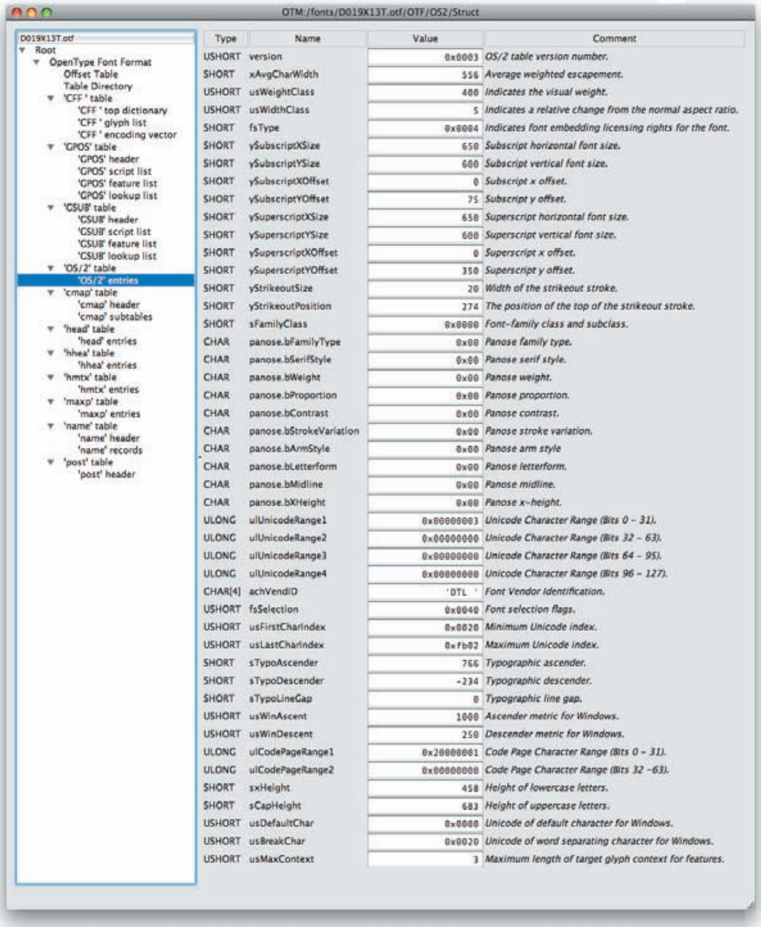
Note: A wws-conformant font may have both Italic and Oblique styles! See Microsoft's paper WPF Font Selection Model. ▶ *www*

Note: If you set the OS/2 *table's version to 4, then all the* ***fsSelection*** *bits 7–9 need to be set to 0 or 1 consciously because starting with this table-version, these bits bear a special meaning.*

— OS/2

Most font-wide information are located in the OS/2 table. The content area's **Comment** column provides a description for every entry.

Info: The OS/2 *table.* ▶ *local* ▶ *www*



Type	Name	Value	Comment
USHORT	version	0x0003	*OS/2 table version number.*
SHORT	xAvgCharWidth	556	*Average weighted escapement.*
USHORT	usWeightClass	400	*Indicates the visual weight.*
USHORT	usWidthClass	5	*Indicates a relative change from the normal aspect ratio.*
SHORT	fsType	0x0004	*Indicates font embedding licensing rights for the font.*
SHORT	ySubscriptXSize	650	*Subscript horizontal font size.*
SHORT	ySubscriptYSize	600	*Subscript vertical font size.*
SHORT	ySubscriptXOffset	0	*Subscript x offset.*
SHORT	ySubscriptYOffset	75	*Subscript y offset.*
SHORT	ySuperscriptXSize	650	*Superscript horizontal font size.*
SHORT	ySuperscriptYSize	600	*Superscript vertical font size.*
SHORT	ySuperscriptXOffset	0	*Superscript x offset.*
SHORT	ySuperscriptYOffset	350	*Superscript y offset.*
SHORT	yStrikeoutSize	20	*Width of the strikeout stroke.*
SHORT	yStrikeoutPosition	274	*The position of the top of the strikeout stroke.*
SHORT	sFamilyClass	0x0000	*Font-family class and subclass.*
CHAR	panose.bFamilyType	0x00	*Panose family type.*
CHAR	panose.bSerifStyle	0x00	*Panose serif style.*
CHAR	panose.bWeight	0x00	*Panose weight.*
CHAR	panose.bProportion	0x00	*Panose proportion.*
CHAR	panose.bContrast	0x00	*Panose contrast.*
CHAR	panose.bStrokeVariation	0x00	*Panose stroke variation.*
CHAR	panose.bArmStyle	0x00	*Panose arm style*
CHAR	panose.bLetterform	0x00	*Panose letterform.*
CHAR	panose.bMidline	0x00	*Panose midline.*
CHAR	panose.bXHeight	0x00	*Panose x-height.*
ULONG	ulUnicodeRange1	0x00000003	*Unicode Character Range (Bits 0 – 31).*
ULONG	ulUnicodeRange2	0x00000000	*Unicode Character Range (Bits 32 – 63).*
ULONG	ulUnicodeRange3	0x00000000	*Unicode Character Range (Bits 64 – 95).*
ULONG	ulUnicodeRange4	0x00000000	*Unicode Character Range (Bits 96 – 127).*
CHAR[4]	achVendID	'DTL '	*Font Vendor Identification.*
USHORT	fsSelection	0x0040	*Font selection flags.*
USHORT	usFirstCharIndex	0x0020	*Minimum Unicode index.*
USHORT	usLastCharIndex	0xfb02	*Maximum Unicode index.*
SHORT	sTypoAscender	766	*Typographic ascender.*
SHORT	sTypoDescender	-234	*Typographic descender.*
SHORT	sTypoLineGap	0	*Typographic line gap.*
USHORT	usWinAscent	1000	*Ascender metric for Windows.*
USHORT	usWinDescent	250	*Descender metric for Windows.*
ULONG	ulCodePageRange1	0x20000001	*Code Page Character Range (Bits 0 – 31).*
ULONG	ulCodePageRange2	0x00000000	*Code Page Character Range (Bits 32 –63).*
SHORT	sxHeight	458	*Height of lowercase letters.*
SHORT	sCapHeight	683	*Height of uppercase letters.*
USHORT	usDefaultChar	0x0000	*Unicode of default character for Windows.*
USHORT	usBreakChar	0x0020	*Unicode of word separating character for Windows.*
USHORT	usMaxContext	3	*Maximum length of target glyph context for features.*

usWeightClass, **usWidthClass** and **fsSelection** relate to the name table records with nameIDs 1/2, 16/17, 21/22, while **fsSelection** also relates to the head table's **macStyle**. Please see the ▶ *name* table section.

fsType defines embedding licensing rights for this font. May it be embedded into a document? What, then, is allowed with such a document?

sTypoAscender, **sTypoDescender**, **sTypoLineGap**, **usWinAscent**, **usWinDescent** – and the hhea table's **ascender**, **descender**, **lineGap** – are dealt with in the ▶ *Consistency Checker* chapter.

If you set the OS/2 table's version to 4, you need to set **fsSelection** bits 7–9 to 0 or 1 consciously because starting with this table version, each of these bits carries a special meaning:

7. To tell applications that the default line-to-line distance should be calculated from **sTypoAscender/sTypoDescender/sTypoLineGap**, set this bit to 1. Otherwise, set this bit to 0. See the ▶ *Consistency Checker* chapter for more information about setting vertical font metrics.

8. If this family is WWS-conformant (the name table *Preferred Subfamily* names (17) address no other categories than weight, width and slope), set this bit to 1. If this is not so (e.g. if *Preferred Subfamily* names (17) refer to optical size or even something like 'Outline'), set this bit to 0 but add WWS-conformant *WWS Family* name (21) and *WWS Subfamily* name (22).

Tip: *See Microsoft's paper* WPF Font Selection Model, *especially pp. 4–6 and the 'Guidelines for font manufacturers' on pp. 17–18.* ▶ *www*

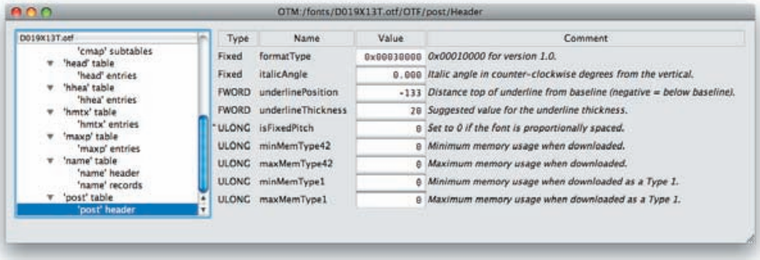
9. If this font's slope can be described as 'Oblique' rather than 'Italic' –WPF distinguishes between 'Oblique' and 'Italic'–, set this bit to 1. Otherwise, set this bit to 0. (If the font is 'Italic', set bit 0 to 1 as usual.)

— post

Info: The post *table.* ▶ *local* ▶ *www*

CFF-based OpenType fonts, unless they are CID-keyed, store glyph names in the CFF table. In this case, the post table has format 3.0 which is a 'short' version that omits glyph name information.

TT-based OpenType fonts store glyph names in a post table of version 2.0. However, it is also possible to not provide glyph names at all, then like in CFF-based OpenType fonts the post has version 3.0.



Type	Name	Value	Comment
Fixed	formatType	0x00030000	*0x00010000 for version 1.0.*
Fixed	italicAngle	0.000	*Italic angle in counter-clockwise degrees from the vertical.*
FWORD	underlinePosition	-133	*Distance top of underline from baseline (negative = below baseline).*
FWORD	underlineThickness	20	*Suggested value for the underline thickness.*
ULONG	isFixedPitch	0	*Set to 0 if the font is proportionally spaced.*
ULONG	minMemType42	0	*Minimum memory usage when downloaded.*
ULONG	maxMemType42	0	*Maximum memory usage when downloaded.*
ULONG	minMemType1	0	*Minimum memory usage when downloaded as a Type 1.*
ULONG	maxMemType1	0	*Maximum memory usage when downloaded as a Type 1.*

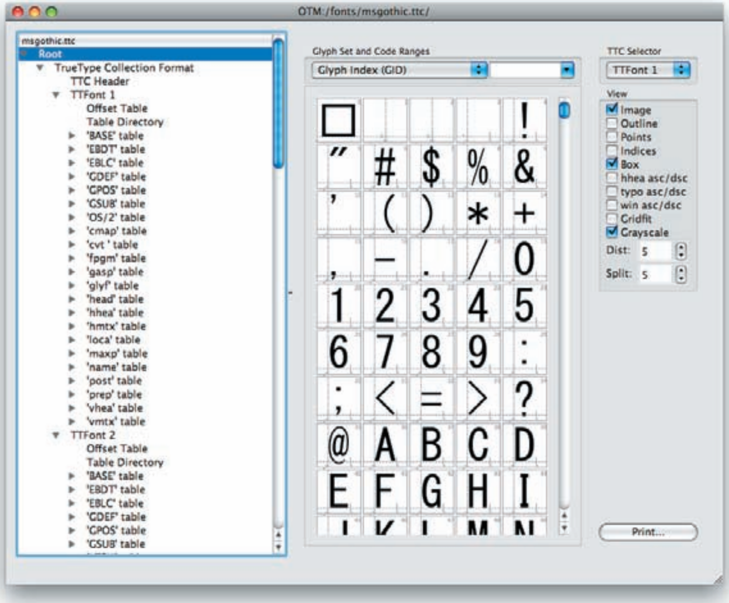
A version 3.0 post *table.*

The **italicAngle** value should be consistent with the **ItalicAngle** in 'CFF' top dictionary (if this is a CFF-based OpenType font) and with the hhea table's **caretSlopeRise** and **caretSlopeRun**. More in the ▶ *Consistency Checker* chapter.

If the font is a CFF-based OpenType font, make sure **underlinePosition** and **underlineThickness** are consistent with the 'CFF' top dictionary's **UnderlinePosition** and **UnderlineThickness**. The underline position is calculated differently in both tables, please see the ▶ *CFF* section for details.

TTC Fonts

A TTC (TrueType Collection) font contains more than one sub-font. A TTC font's TTC Header points to each sub-font's header information consisting of Offset Table and Table Directory which points to tables associated with this sub-font. (Version 2.0 of the TTC Header will also reference a common DSIG table.) These then are followed by all tables. This allows a TTC's sub-fonts to 'share' tables that are identical, and – which is the example given in the OpenType specification – even use a single glyf table holding all glyphs of all sub-fonts, yet each sub-font's loca table only refers to glyphs relevant for this sub-font.



Glyph Set and Code Ranges: Glyph Index (GID)

TTC Selector: TTFont 1

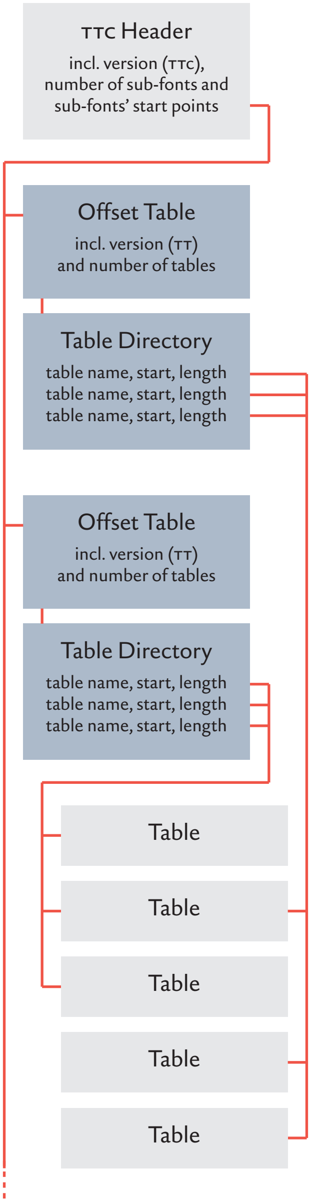
View:
☑ Image
☐ Outline
☐ Points
☐ Indices
☑ Box
☐ hhea asc/dsc
☐ typo asc/dsc
☐ win asc/dsc
☐ Gridfit
☑ Grayscale
Dist: 5
Split: 5

With TTC fonts, OTMaster's table overview starts with the Root entry. Root holds TrueType Collection Format which in turn holds TTFont 1, TTFont 2, TTFont 3, etc., as many as there are sub-fonts in the TTC font. Each TTFont (like the OpenType Font Format) holds an Offset Table and Table Directory, followed by all tables found in the sub-font.

OTMaster lists, for each sub-font, all referenced tables – even if these are 'shared' by sub-font. Once that a 'shared' table is adjusted for a single font, this table will be duplicated.

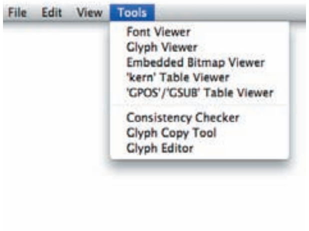
Also see *The OpenType Font File.* ▶ *local* ▶ *www*

The structure of a TTC font's data (simplified):


TTC Header
incl. version (TTC), number of sub-fonts and sub-fonts' start points
Offset Table
incl. version (TT) and number of tables
Table Directory
table name, start, length
table name, start, length
table name, start, length
Offset Table
incl. version (TT) and number of tables
Table Directory
table name, start, length
table name, start, length
table name, start, length
Table
Table
Table
Table
Table

OTMaster's Toolbox

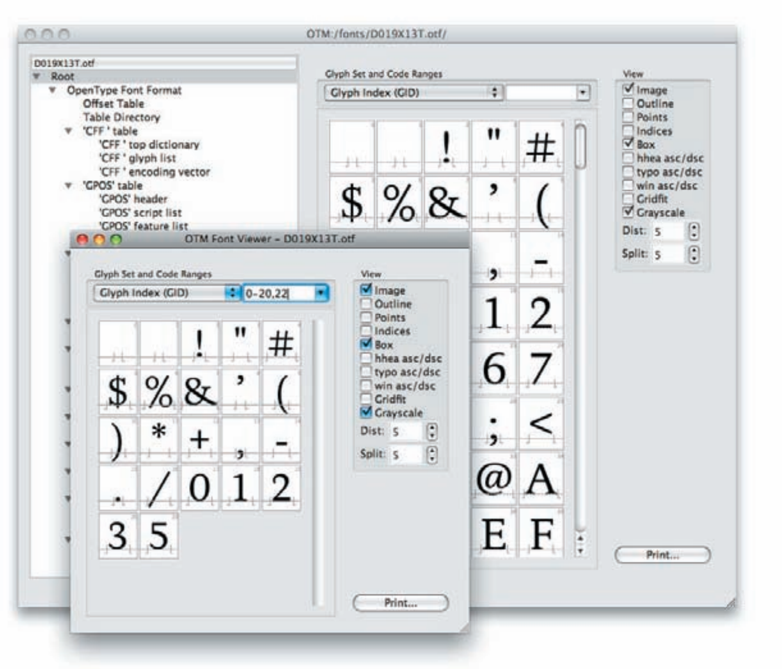
OTMaster is not only good for reviewing and editing a font's 'raw' tables but comes with tools for higher-level – and thus more user-friendly – access to parts of a font's data. Some of these tools do not show a specific table's content but represent data in a thematic/categorized way: The Font Viewer shows all, or a selection, of a font's glyphs. The Glyph Viewer shows all data related to an individual glyph, thereby combining data from various tables like glyf or CFF for outlines, post or CFF for glyph names, cmap for Unicode codepoints, hmtx and/or vmtx for metrics, etc.



TOOLS MENU

Font Viewer

The table overview's Root and the Font Viewer alike give a visual summary of a font's glyph set:



Glyph Index (GID) 0-20,22

☑ Image
☐ Outline
☐ Points
☐ Indices
☑ Box
☐ hhea asc/dsc
☐ typo asc/dsc
☐ win asc/dsc
☐ Gridfit
☑ Grayscale
Dist: 5
Split: 5

When selecting Root, its content area shows the same glyph overview as does the Font Viewer.

— Glyph Set and Code Ranges

These define which part of the glyph set to show, and how:

The leftside popup menu offers categories for defining glyph sub-sets by **Glyph Index (GID)**, **Unicode**, or any cmap subtable present in the font. The category also determines how glyphs are sorted in the overview.

The rightside textbox allows you to restrict the glyph overview to selected glyphs or glyph ranges. These are defined by GID or Unicode codepoint, depending on the category selected in the popup to the left.

*Glyph indices are expected to be integer numbers (0, 1, 123). Unicode codepoints are expected to be hexadecimal numbers, without trailing zeros but preceded by '**0x**' (0x31, 0xB5).*

You can provide a single identifier (0x20), a comma-separated list of identifiers (0x20, 0x61), a range delimited by first and last identifier (0x20-0x61), or a combination thereof (0x20, 0x40-0x61, 0xB5).

You need to confirm with ⏎.

The text input box will remember previous glyph sub-sets, and you may get back to them by using the boxes' popup functionality – by clicking on the popup menu's arrow button or by placing the cursor inside the box and using ↑ or ↓ to jump to any previous glyph sub-set.

— **View**

Image shows filled shapes.
Outline adds a colored outline.
Points shows on-curve points.
Indices shows points' index numbers.
Box draws the bounding box around each glyph.
hhea asc/dsc are the hhea table's ascender/descender heights.
typo asc/dsc are the OS/2 table's sTypoAscender/sTypoDescender height.
win asc/dsc are the OS/2 table's usWinAscent/usWinDescent heights.
Gritfit will align points, at current ppem size, to the pixel grid.
Grayscale draws grayscaled outlines.
Dist. is the distance between bounding boxes. With a value of 0, bounding boxes will touch.
Split determines the number of glyphs per line.

Note: Since OTMaster uses the FreeType rasterizer, ***Grayscale*** *may produce results that differ from 'real world' output, especially at small ppem sizes.*

A glyph index is shown in the top left corner of each glyph's bounding box.

A double-click on a glyph in the Font Viewer glyph overview, or in the Root content area, will open this glyph in the ▶ *Glyph Viewer*.

⇧ + double-click on a glyph will open it in the ▶ *Glyph Editor*.

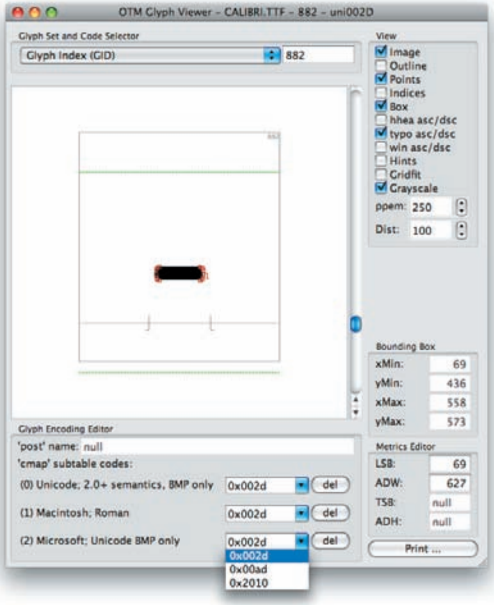
Glyph Viewer

Per glyph, the Glyph Viewer interface gives access to all glyph-related data.

— Glyph Set and Code Selector

In the left popup menu choose whether to identify a glyph by **Glyph Index (GID)** or **Unicode** codepoint. In the right text input box, define a glyph's glyph index or codepoint.

*Glyph indices are expected to be integer numbers (0, 1, 123). Unicode codepoints are expected to be hexadecimal numbers, without trailing zeros but preceded by '**0x**' (0x31, 0xB5).*



OTM Glyph Viewer – CALIBRI.TTF – 882 – uni002D

Glyph Set and Code Selector
Glyph Index (GID): 882

View
☑ Image
☐ Outline
☑ Points
☐ Indices
☑ Box
☐ hhea asc/dsc
☑ typo asc/dsc
☐ win asc/dsc
☐ Hints
☐ Gridfit
☑ Grayscale
ppem: 250
Dist: 100

Bounding Box
xMin: 69
yMin: 436
xMax: 558
yMax: 573

Glyph Encoding Editor
'post' name: null
'cmap' subtable codes:
(0) Unicode; 2.0+ semantics, BMP only: 0x002d
(1) Macintosh; Roman: 0x002d
(2) Microsoft; Unicode BMP only: 0x002d

Metrics Editor
LSB: 69
ADW: 627
TSB: null
ADH: null

Use the scrollbar to the right of the glyph review area to flip through all glyphs!

— View

Image shows filled shapes.
Outline adds a colored outline.
Points shows on-curve points.
Indices shows points' index numbers.
Box draws the bounding box around each glyph.
hhea asc/dsc are the hhea table's ascender/descender heights.
typo asc/dsc are the OS/2 table's sTypoAscender/sTypoDescender height.
win asc/dsc are the OS/2 table's usWinAscent/usWinDescent heights.
Hints shows hints if existing.
Gritfit will align points, at current ppem size, to the pixel grid.
Grayscale draws grayscaled outlines.
ppem is the ppem size at which the glyph is rasterized.
Zoom allows you to scale the rasterized image up or down which makes it possible to see the actual rasterization in detail.

*Note: Since OTMaster uses the FreeType rasterizer, **Grayscale** may produce results that differ from 'real world' output, especially at small ppem sizes.*

*Tip: If a font's upm (units per em) is 1000 and if you choose 160 **ppem**, then the 1000 units will be scaled such that they fit into 160 pixels upon glyph rasterization. The effect of this can be studied by increasing the **Zoom** factor.*

— Glyph Encoding Editor

The first entry is the glyph name, which is called
1. **'CFF' id** with CFF-based OpenType fonts ('id' because in CID-keyed fonts this would not be a name but a mere index value), and
2. **'post' name** with TT-based OpenType fonts.
'cmap' subtable codes shows as many entries as there are subtables in the cmap table. For each subtable there is a popup text box which may contain none, one or more Unicode codepoints. Click the **del** button to delete the respective subtable's codepoints.

Changes in this area will be reflected in 'cmap' table and vice versa.

Tip: Make sure that you provide or adjust codepoints for all cmap *subtables, not just one …*

— Bounding Box

xMin is the smallest horizontal extension of the glyph.
yMin is the smallest vertical extension of the glyph.
xMax is the largest horizontal extension of the glyph.
yMax is the largest vertical extension of the glyph.

Note: In CFF-based OpenType fonts, these values relate to smallest or largest extensions of a glyph regardless if there are on-curve extremum points or not. In TT-based OpenType fonts, these values relate to coordinates of outermost on-curve or off-curve points.

— Metrics Editor

LSB is the left sidebearing which is equal to xMin.
ADW is the advance width.
TSB is the top bearing.
ADH is the advance height.

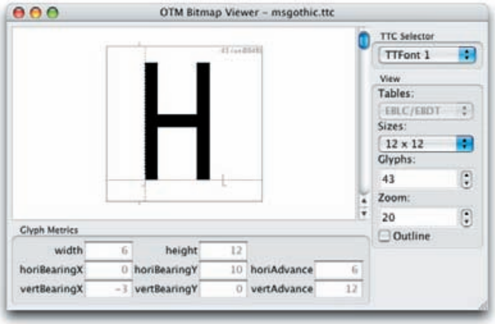
The right sidebearing is implicit and can be calculated as

RSB = ADW – xMax

TSB and **ADH** relate to vertically oriented glyphs whose origin point, in CFF-based OpenType fonts, is at the top center of a glyph's bounding box. An illustrations for this can be found in Adobe's AFM specifications, p. 11, fig. 1. ▶*www*

Embedded Bitmap Viewer

The Embedded Bitmap Viewer visualizes bitmaps from EBLC and EBDT or other bitmap tables.



OTM Bitmap Viewer – msgothic.ttc

TTC Selector: TTFont 1

View

Tables: EBLC/EBDT

Sizes: 12 x 12

Glyphs: 43

Zoom: 20

☐ Outline

Glyph Metrics

width 6

height 12

horiBearingX 0

horiBearingY 10

horiAdvance 6

vertBearingX -3

vertBearingY 0

vertAdvance 12

The Embedded Bitmap Viewer. Use the scrollbar to the right of the glyph review area to flip through all glyphs!

— View

Tables allows you to select which tables' bitmap information to show.
Sizes are the bitmap sizes covered by the selected tables.
Glyphs allows choosing a glyph by its glyph index.
Zoom will zoom in or out.
Outline displays the outline as well.

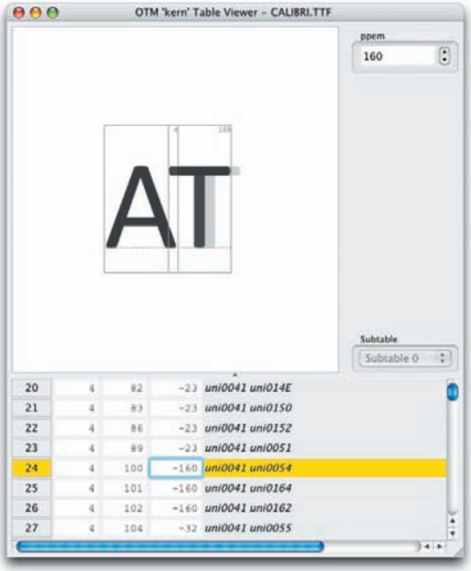
— Glyph Metrics

width is the number of columns of data, i.e. the number of pixels in horizontal direction.
height is the number of rows of data, i.e. the number of pixels in vertical direction.
horiBearingX is the number of pixels from the origin to the left edge of the bitmap (horizontal layout).
horiBearingY is the number of pixels from the origin to the top edge of the bitmap (horizontal layout).
horiAdvance is the bitmap's advance width (horizontal layout).
vertBearingX is the number of pixels from the origin to the left edge of the bitmap (vertical layout).
vertBearingY is the number of pixels from the origin to the top edge of the bitmap (vertical layout).
vertAdvance is the bitmap's advance height (vertical layout).

All these values are read-only.

‘kern’ Table Viewer

This is a most useful tool to visually check and adjust the kern table. The currently selected pair is presented in main part of the window.



OTM 'kern' Table Viewer - CALIBRI.TTF

ppem: 160

Subtable: Subtable 0

20	4	82	-23	uni0041 uni014E
21	4	83	-23	uni0041 uni0150
22	4	86	-23	uni0041 uni0152
23	4	89	-23	uni0041 uni0051
24	4	100	-160	uni0041 uni0054
25	4	101	-160	uni0041 uni0164
26	4	102	-160	uni0041 uni0162
27	4	104	-32	uni0041 uni0055

The ‘kern’ Table Viewer dialog.

— ppem

This single option serves to enlarge or reduce the size of the kerning pair, by choosing another ppem size.

— Subtable

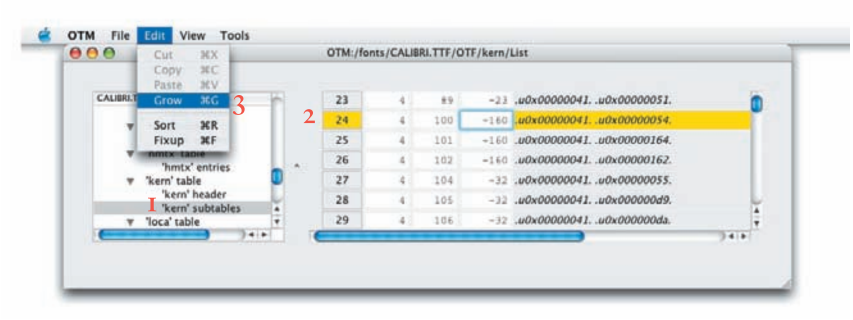
Here you may select which subtable’s kerning you want to review or edit.

Note: A kern table may contain more than one subtable, but please be aware that Windows and OS/2 accept a kern table only if it holds a single subtable of format 0. Also see the kern table specification. ▶ local ▶ www

The list of kerning pairs which follows consists of three editable columns:
left glyph of the pair by glyph index,
right glyph of the pair by glyph index,
and the kerning **value** which is relative to the font’s upm value.
In addition, a **Comment** column translates left and right glyphs’ indices into glyph names as found in either CFF or post table.

Like in **Nested Tables** viewing mode, use ↑ (arrow up) and ↓ (arrow down) keys to flip through all kerning pairs quickly, and ⇥ (to right) and ⇧+⇥ (or ⇤; to left) to jump from column to column.

Existing kerning pairs – **left** and **right** glyphs' indices as well as the kerning **value** – can be reviewed and adjusted in the 'kern' Table Viewer, but it is not possible to remove or add kerning pairs. With a little trick, though, you may do so:



23	4	89	-23	.u0x00000041. .u0x00000051.
24	4	100	-160	.u0x00000041. .u0x00000054.
25	4	101	-160	.u0x00000041. .u0x00000164.
26	4	102	-160	.u0x00000041. .u0x00000162.
27	4	104	-32	.u0x00000041. .u0x00000055.
28	4	105	-32	.u0x00000041. .u0x000000d9.
29	4	106	-32	.u0x00000041. .u0x000000da.

Remove or add a kerning pair in three steps.

1. Go to the main window and select 'kern' subtables (inside of 'kern' table) from the table overview. Click the **+** sign next to the subtable which you intend to edit, this will fold out the subtable as a nested table. Possibly double-click the subtable's header to show its entries in a new window.

Now either remove a kerning pair:

2. Select the pair which you intend to remove.
3. Choose **Edit > Cut** (⌘+X) (CTRL+X).

Or add a new kerning pair:

2. Select any kerning pair, preferrably the last one.
3. Choose **Edit > Grow** (⌘+G) (CTRL+G) to duplicate this selected pair.

Change **left** glyph index, **right** glyph index and/or kerning **value**, and you have added a new kerning pair! Go back to the 'kern' Table Viewer and visually adjust the kerning **value**. If you added new pair(s) at the end of the kern table, you know where to find them …

Finally choose **Edit > Fixup** (⌘+F) (CTRL+F) to reorder kerning pairs by glyph indices.

‘GPOS’/‘GSUB’ Viewer

The ‘GPOS’/‘GSUB’ Viewer is a tool for reviewing the content of the two most important OpenType layout tables, GSUB for glyph substitution and GPOS for glyph positioning. In case of the GPOS table, positioning values can be adjust. The viewer presents these tables’ data visually, which makes it possible to check a font’s layout behavior in a comfortable way. Data is, structurally, presented in the same way as found in these tables.

Both GSUB and GPOS tables consist of three lists: Script list, Feature list, and Lookup list.

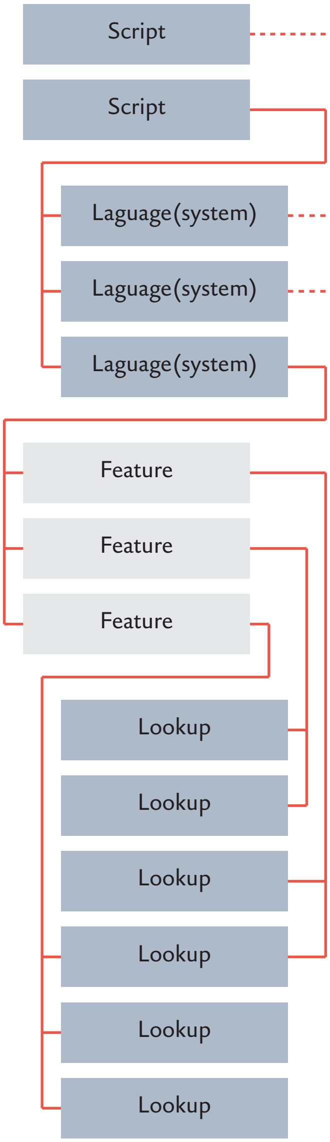
The Script list sums up all scripts explicitly addressed by the layout table (these are scripts – or writing systems – like ‘latn’ for Latin or ‘cyrl’ for Cyrillic).* Per each script, there is a list of languages explicitly addressed by the layout table as well as a ‘default’ language for which there is a special place in the data structure. In case that a layout application cannot find a match for the selected language (e.g. by way of the spelling dictionary in Adobe InDesign) in the font’s layout table, it would fall back to this ‘default’ language. And per each language there is a list of features associated with it.

The Feature list, for each feature to which a script/language combination refers, points to one or more lookups in which the actual substitution and positioning behavior is defined.

Finally, the Lookup list points to individual lookups, each of which defines portions of layout behavior.

Users of VOLT are familiar with this structure because VOLT presents layout data in a way which resembles layout tables’ data structure. Users of AFDKO and Adobe’s feature file syntax may need some time to get accustomed to it because the higher-level nature of Adobe’s feature file syntax hides the complexity of the data structure of compiled layout tables. So again our recommendation that you study especially the documents related to OpenType layout tables, in particular to GSUB and GPOS.

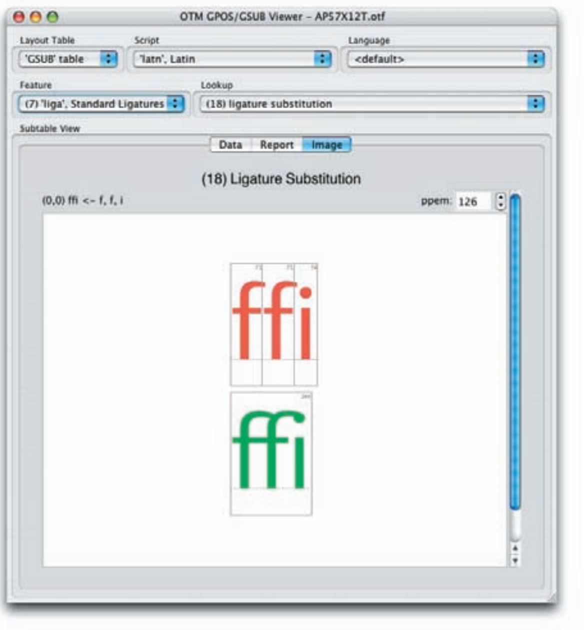
OpenType layout data is organized by script, language system, typographic feature and lookup:



** The default script, or ‘DFLT’ (all-caps!), was introduced rather late by Adobe. While there is a special place for the default language, or ‘dflt’ (lowercase!), in the Language list, there is no such thing for the ‘DFLT’ script – it is a script like any other, part of the regular Script list, and identified by its tag.*

Info: The GPOS *table.* ▶ *local* ▶ *www The* GSUB *table.* ▶ *local* ▶ *www Also see the* OpenType Layout Common Table Formats document. ▶ *local* ▶ *www*

The ‘GPOS’/‘GSUB’ Viewer’s top popup boxes reflect the layout tables’ structure:



OTM GPOS/GSUB Viewer – AP57X12T.otf

Layout Table: 'GSUB' table

Script: 'latn', Latin

Language: <default>

Feature: (7) 'liga', Standard Ligatures

Lookup: (18) ligature substitution

Subtable View: Image

(18) Ligature Substitution

(0,0) ffi <- f, f, i

ppem: 126

— Layout Table

At first you need to choose from ‘GSUB’ table or ‘GPOS’ table – ‘GSUB’ table cares for glyph substitution and ‘GPOS’ table cares for glyph positioning.

— Script

Select the script whose lookups you plan to review.

— Language

Select the language whose lookups you plan to review. The choise of languages depends on which **Script** you have selected previously.

— Feature

Select a feature whose lookups you plan to review. The choice features depends on which **Language** you have selected previously.

— Lookup

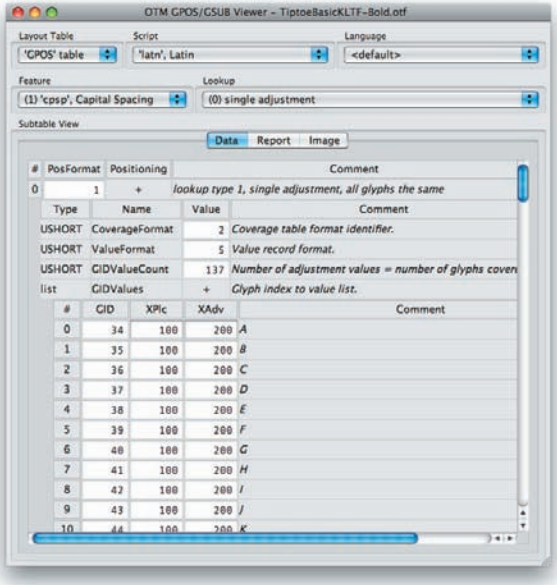
The selection of lookups shown in this popup box is determined by your previous choices of **Script**, **Language** and **Feature**.

Tip: To see more lookups, and independently of previous choices of ***Script, Language*** *and/or* ***Feature,*** *select the option* <any> *in* ***Script, Language*** *and/or* ***Feature.***

— Subtable View

The content of the lookup can be viewed in three modes.

1. **Data**. This is exactly what you see if you inspect 'GSUB' lookups or 'GPOS' lookups in the table content area:



OTM GPOS/GSUB Viewer – TiptoeBasicKLTF-Bold.otf

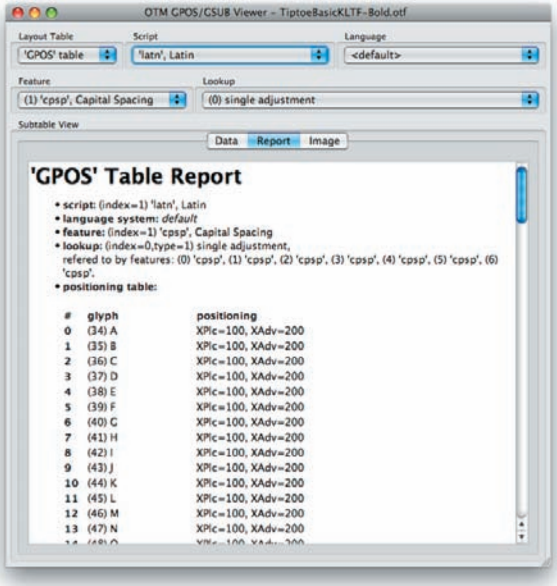
Layout Table: 'GPOS' table
Script: 'latn', Latin
Language: <default>
Feature: (1) 'cpsp', Capital Spacing
Lookup: (0) single adjustment

#	PosFormat	Positioning	Comment
0	1	+	*lookup type 1, single adjustment, all glyphs the same*

Type	Name	Value	Comment
USHORT	CoverageFormat	2	*Coverage table format identifier.*
USHORT	ValueFormat	5	*Value record format.*
USHORT	GIDValueCount	137	*Number of adjustment values = number of glyphs coven*
list	GIDValues	+	*Glyph index to value list.*

#	GID	XPlc	XAdv	Comment
0	34	100	200	*A*
1	35	100	200	*B*
2	36	100	200	*C*
3	37	100	200	*D*
4	38	100	200	*E*
5	39	100	200	*F*
6	40	100	200	*G*
7	41	100	200	*H*
8	42	100	200	*I*
9	43	100	200	*J*
10	44	100	200	*K*

2. **Report**. Provides you with a text report of the selected lookup's content:



OTM GPOS/GSUB Viewer – TiptoeBasicKLTF-Bold.otf

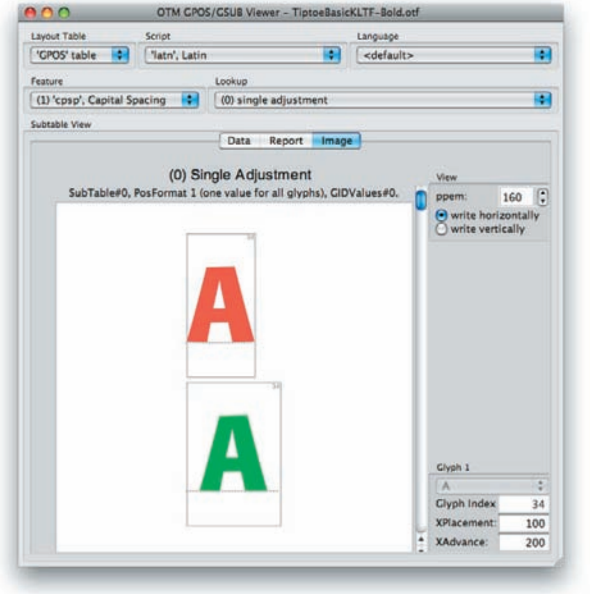
Layout Table: 'GPOS' table
Script: 'latn', Latin
Language: <default>
Feature: (1) 'cpsp', Capital Spacing
Lookup: (0) single adjustment

'GPOS' Table Report

- **script:** (index=1) 'latn', Latin
- **language system:** *default*
- **feature:** (index=1) 'cpsp', Capital Spacing
- **lookup:** (index=0,type=1) single adjustment,
  refered to by features: (0) 'cpsp', (1) 'cpsp', (2) 'cpsp', (3) 'cpsp', (4) 'cpsp', (5) 'cpsp', (6) 'cpsp'.
- **positioning table:**

```
#   glyph        positioning
0   (34) A       XPlc=100, XAdv=200
1   (35) B       XPlc=100, XAdv=200
2   (36) C       XPlc=100, XAdv=200
3   (37) D       XPlc=100, XAdv=200
4   (38) E       XPlc=100, XAdv=200
5   (39) F       XPlc=100, XAdv=200
6   (40) G       XPlc=100, XAdv=200
7   (41) H       XPlc=100, XAdv=200
8   (42) I       XPlc=100, XAdv=200
9   (43) J       XPlc=100, XAdv=200
10  (44) K       XPlc=100, XAdv=200
11  (45) L       XPlc=100, XAdv=200
12  (46) M       XPlc=100, XAdv=200
13  (47) N       XPlc=100, XAdv=200
```

3. **Image**. This is the default viewing mode. Substitution or re-positioning is visualized, so that you may evaluate a lookup's layout behavior quickly at a glance, and even evaluate positioning adjustments:


OTM GPOS/GSUB Viewer – TiptoeBasicKLTF-Bold.otf

Layout Table: 'GPOS' table
Script: 'latn', Latin
Language: <default>
Feature: (1) 'cpsp', Capital Spacing
Lookup: (0) single adjustment

(0) Single Adjustment
SubTable#0, PosFormat 1 (one value for all glyphs), GIDValues#0.

ppem: 160
● write horizontally
○ write vertically

Glyph 1: A
Glyph Index: 34
XPlacement: 100
XAdvance: 200

Use the scrollbar to the right of the preview area to flip through all substitutions or positioning adjustments.

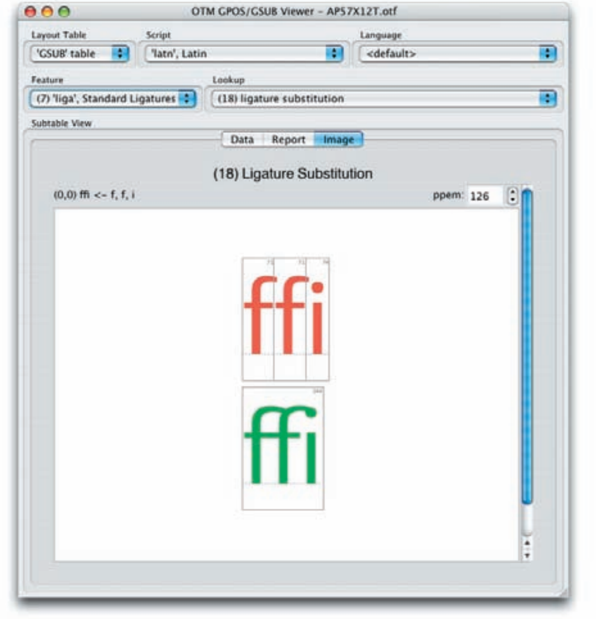
The main area is headed by this lookup's index (in parentheses) and type, followed by additional information like the index of the current subtable or the substitution or positioning format. The original glyph is colored RED, the replaced or repositioned glyph is colored GREEN.

Like in other OTMaster dialogs, use the scrollbar next to the review image to switch from one substitution or positioning entry to another!

Which information are shown in **Image** viewing mode depends on whether you are inspecting 'GSUB' table or 'GPOS' table and also depends on the lookup type. A few examples are given below.

1. GSUB:

1.1. With non-contextual substitution, there are no further options. You may use the scrollbar to flip through glyph substitutions:


OTM GPOS/GSUB Viewer – AP57X12T.otf

Layout Table: 'GSUB' table

Script: 'latn', Latin

Language: <default>

Feature: (7) 'liga', Standard Ligatures

Lookup: (18) ligature substitution

Subtable View: Image

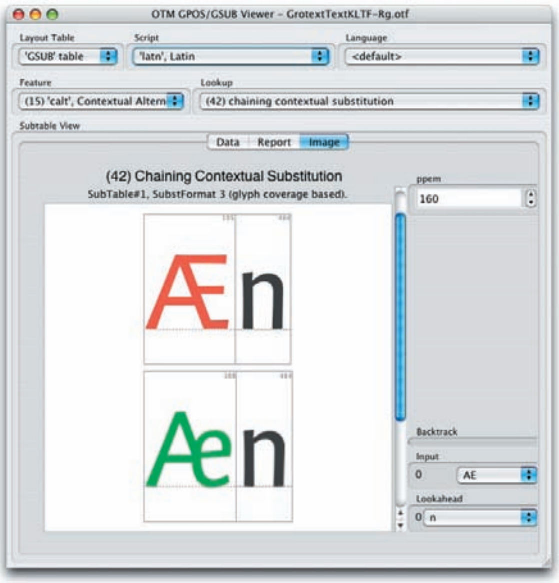
(18) Ligature Substitution

(0,0) ﬃ <- f, f, i

ppem: 126

ppem is the ppem size at which the review image is rasterized.

1.2. With contextual substitution, there are a few more pieces of information. Besides a RED original glyph and a GREEN adjusted glyph, there is a BLACK context glyph in the preview area:



OTM GPOS/GSUB Viewer – GrotextTextKLTF-Rg.otf

Layout Table: 'GSUB' table
Script: 'latn', Latin
Language: <default>
Feature: (15) 'calt', Contextual Altern
Lookup: (42) chaining contextual substitution

(42) Chaining Contextual Substitution
SubTable#1, SubstFormat 3 (glyph coverage based).

ppem: 160
Input: 0 AE
Lookahead: 0 n

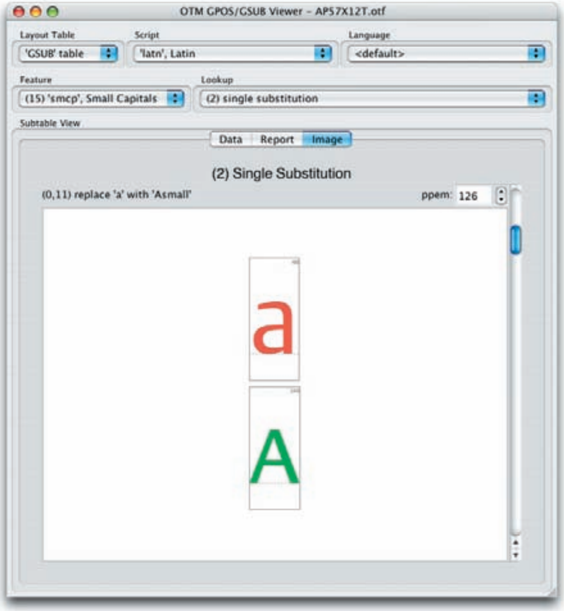
— **ppem**
The size at which the review image is rasterized.

— **Backtrack**
— **Input**
— **Lookahead**
Input is always shown, this is the glyph or glyphs will be substituted, together with **Backtrack** and/or **Lookahead**. Contextual substitution may involve more than just one of each!

If (in the order of appearance) **Backtrack** or **Input** or **Lookahead** involves but a single glyph, the respective popup is greyed out and shows the single glyph's name. If **Backtrack** or **Input** or **Lookahead** involves a glyph class, then the popup shows the first glyph by default. You may use the popup to select any other glyph: just click on the popup and move the mouse up or down the list, and the review will instantly show the glyph touched by the mouse – no need to click.

2. GSUB:

2.1. With single adjustment, the affected glyph as well as the adjustment value are shown:


OTM GPOS/GSUB Viewer – AP57X12T.otf

Layout Table: 'GSUB' table
Script: 'latn', Latin
Language: <default>
Feature: (15) 'smcp', Small Capitals
Lookup: (2) single substitution
Subtable View: Image

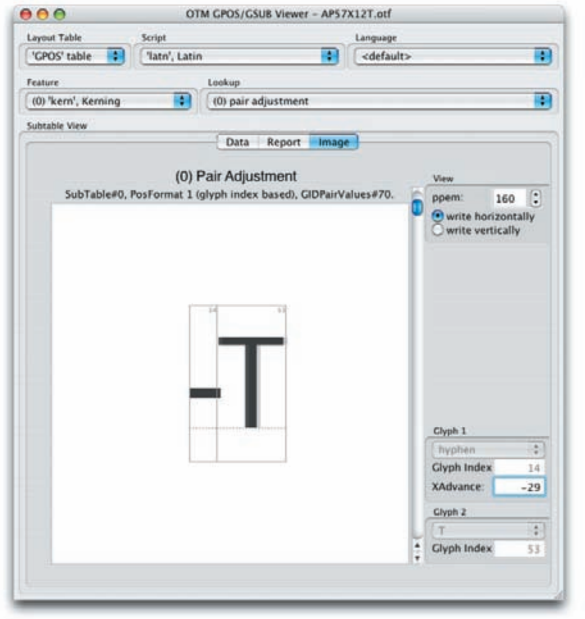
(2) Single Substitution

(0,11) replace 'a' with 'Asmall'

ppem: 126

ppem is the ppem size at which the review image is rasterized.

2.2. With pair adjustment, there are a few additions:



OTM GPOS/GSUB Viewer – AP57X12T.otf

Layout Table: 'GPOS' table
Script: 'latn', Latin
Language: <default>
Feature: (0) 'kern', Kerning
Lookup: (0) pair adjustment
Subtable View: Data | Report | Image

(0) Pair Adjustment
SubTable#0, PosFormat 1 (glyph index based), GIDPairValues#70.

View
ppem: 160
● write horizontally
○ write vertically

Glyph 1
hyphen
Glyph Index 14
XAdvance: -29

Glyph 2
T
Glyph Index 53

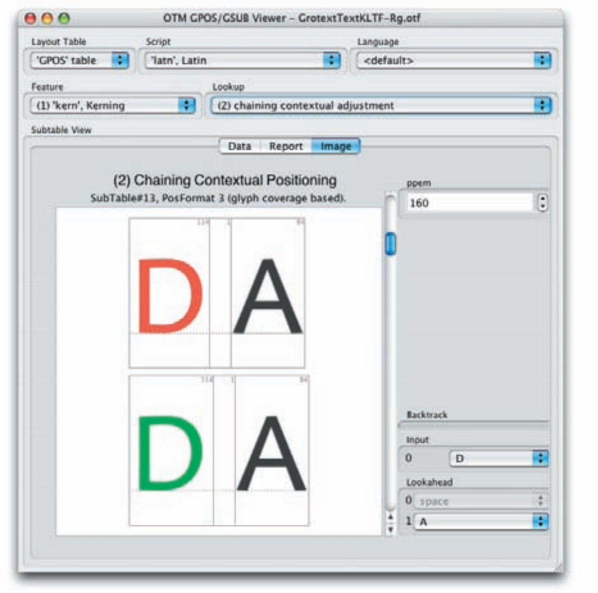
— View

ppem is the ppem size at which the review image is rasterized.
In addition, two radio buttons allow to choose from **write horizontally** and **write vertically** which will arrange the glyphs involved either side by side or one above the other.

— Glyph 1
— Glyph 2

For each of them there is a popup with the glyph name. If it is but a single glyph, the popup will be greyed out. If it is a glyph class, then you may click on the popup and move the mouse up or down the list to see any of the other glyphs in the review – no need for clicking. **Glyph Index** is the glyph index of the glyph currently shown. Depending on which kind of adjustment has been made, values for **XPositioning**, **XAdvance**, **YPositioning** and/or **YAdvance** adjustment are shown. The latter values can be adjusted. However, the general structure of GSUB and GPOS tables cannot be changed: it is not possible to add anything that has not been in these tables before, nor to remove anything.

2.3. With contextual positioning, BLACK context glyphs join RED original glyph and GREEN adjusted glyph:


OTM GPOS/GSUB Viewer – GrotextTextKLTF-Rg.otf

Layout Table: 'GPOS' table
Script: 'latn', Latin
Language: <default>
Feature: (1) 'kern', Kerning
Lookup: (2) chaining contextual adjustment
Subtable View: Image

(2) Chaining Contextual Positioning
SubTable#13, PosFormat 3 (glyph coverage based).

ppem: 160
Backtrack
Input: 0 D
Lookahead: 0 space; 1 A

— ppem

The size at which the review image is rasterized.

— Backtrack
— Input
— Lookahead

Input is always shown, this is the glyph or glyphs to be substituted, together with **Backtrack** and/or **Lookahead**.

No values are shown, but the effect is visible in the preview. In the example above, the 'D's right sidebearing is increased when followed by space and a class which contains an 'A'.

Note: The 'GSUB'/'GPOS' Viewer allows you to review individual *lookups. If a feature refers to more than one lookups, these lookups' behavior will be additive. This has different effects with* GSUB *and* GPOS.
GSUB: *All lookups associated with a feature will be applied. However, if a previous lookup has substituted an input glyph already, additional lookups will not find a match for the same input glyph any more because this has been substituted already and does not exist any more in the input string. In so far, lookup order matters, as does the order of substitutions inside of a subtable.*
GPOS: *And again, all lookups associated with a feature will be applied. If more than one lookup adjust positioning and/or advance width of a specific glyph, then all lookups' adjustments will* add up. *It is important to keep this in mind, because the 'GSUB'/'GPOS' Viewer visualizes only individual lookups' adjustments, i.e. does not show the result of a feature's total positioning adjustments!*

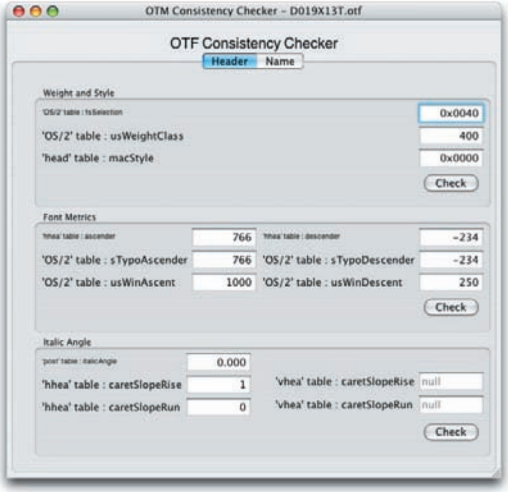
Consistency Checker

This tool helps finding inconsistencies across OpenType fonts' tables.

Tip: In addition, consider using font checking tools like Microsoft's Font Validator ▶ www as well as the command-line tool CompareFamily ▶ www which is included in Adobe's AFDKO.

— Header

The Header section of the Consistency Checker gives an overview, in one dialog, of different table's entries which need to be consistent:



Weight and Style
'OS/2' table : fsSelection 0x0040
'OS/2' table : usWeightClass 400
'head' table : macStyle 0x0000

Font Metrics
'hhea' table : ascender 766
'hhea' table : descender -234
'OS/2' table : sTypoAscender 766
'OS/2' table : sTypoDescender -234
'OS/2' table : usWinAscent 1000
'OS/2' table : usWinDescent 250

Italic Angle
'post' table : italicAngle 0.000
'hhea' table : caretSlopeRise 1
'hhea' table : caretSlopeRun 0
'vhea' table : caretSlopeRise null
'vhea' table : caretSlopeRun null

1. **Weight and Style** compares the OS/2 table's **fsSelection**, **usWeightClass** and the head table's **macStyle**. If a font is a bold style, this should be reflected in all three entries, if a font is an italic style, this should be reflected in **fsSelection** and **macStyle**.

Please see the OS/2 table spec ▶ *local* ▶ *www* for **fsSelection** and **usWeightClass** with the head table spec ▶ *local* ▶ *www* for **macStyle**.

2. **Font Metrics** compares the hhea table's **ascender/descender** with the OS/2 table's **sTypoAscender/sTypoDescender** and **usWinAscent/usWinDescent**. (The hhea table's **linegap** and the OS/2 table's **sTypoLineGap** are not shown in the Consistency Checker.) Please review **Check**'s adjustments – you may not agree with them because it sets:

ascender = sTypoAscender = usWinAscent
descender = sTypoDescender = usWinDescent

Both upm and the three sets of vertical metrics

a. head **unitsPerEm**
b. hhea **ascender/descender/lineGap**
c. OS/2 **sTypoAscender/sTypoDescender/sTypoLineGap**
d. OS/2 **usWinAscent/usWinDescent**

need to be defined carefully because different applications pick different entries to determine the default line-to-line distance. To achieve consistent default line-to-line distance in most – though not in all – applications, you need to take care that the 'sum' of each of the first three sets is the same. Microsoft's and Adobe's current practice for determining (vertical) font metrics –which may be interpreted as a recommendation– is this:

An example that meets all conditions:
head
upm = 1000
OS/2 hhea
sTypoAscender = *ascender* = 800
sTypoDescender = *descender* = -200
sTypoLineGap = *lineGap* = 200
usWinAscent = 850
usWinDescent = 350

2.1 The pair **sTypoAscender/sTypoDescender** indicates how much of upm (the head table's **unitsPerEm**) is reserved for ascenders and descenders. This can be expressed as:

sTypoAscender - sTypoDescender = upm

It is expected that your typeface is designed and upm is defined such that normal ascenders and descenders as of 'A'–'Z' and 'a'–'z' remain within the upm's boundaries, or sTypoAscender and sTypoDescender.

2.2 **sTypoAscender/sTypoDescender/sTypoLineGap** (set c) defines a font's ideal line-to-line distance. It depends on the typeface's design, but usually **sTypoLineGap** is about 20% of:

2.2.1 sTypoAscender - sTypoDescender
2.2.2 upm

(According to 2.1, 2.2.1 and 2.2.2 are equal.) For example, upm = 1000 and sTypoLineGap = 200 result in a default line-to-line distance of 120% of upm which would translate into 10/12 pt.

Tip: *Please consult John Hudson's* Setting Cross-Platform Vertical Metrics, *especially the latest 'Update'.* ►*www Rather for the illustration than the method itself which is obsolete now, see Karsten Lücke's* Font Metrics. ►*www*

2.3 Since **sTypoAscender/sTypoDescender/sTypoLineGap** (c) were defined such that the 'sum' results in an ideal default line-to-line distance, you may set the OS/2 table's version to 4 and **fsSelection** bit 7 to 1. This indicates that **sTypo**-values should be used for calculating default line-to-line distance rather than **usWin**-values. Once the OS/2 table's version is 4 however, you need to set **fsSelection** bits 8 and 9 consciously!

Note: *This is a recommendation, not a requirement! Also see* ►os/2.

2.5 **sTypoAscender/sTypoDescender/sTypoLineGap** (c) and **ascender/descender/lineGap** (b) share same values, so you can simply reuse the first set's values for the latter set:

ascender = sTypoAscender
descender = sTypoDescender
lineGap = sTypoLineGap

2.6 **usWinAscent/usWinDescent** (d) provide information about a font's clipping zones – the largest extensions that you do not want to see clipped, i.e. cut off. These values reflect the tallest glyphs in the font.

Note: *The* hhea *table's **descender** & the* OS/2 *table's **sTypoDescender** are given as negative values – but **usWinDescent** must be given as a positive value! Therefore, the term 'sum' ist not exact in a strict sense.*

With a typeface of normally sized ascenders and descenders, the 'sum' of the **usWin** set will be smaller than the 'sum' of the **sTypo** set. Increase **usWin**-values a bit so that the 'sum' of each of the three sets is identical:

sTypoAscender - sTypoDescender + sTypoLineGap (c)
= usWinAscent + usWinDescent (d)
= ascender - descender + lineGap (b)

In case that a font contains excessively tall glyphs it may be impossible to achieve this equation.

It is strongly recommended that you define vertical metrics such that they are identical across all fonts that belong to a family. This to make sure that if a user relies on default line-to-line distance, this would not vary if one paragraph is set in Regular, the other in Italic or Bold style.

3. **Italic Angle** allows the comparison of the post table’s **italicAngle** with the hhea table’s **caretSlopeRise** and **caretSlopeRun**. The relation between these two is

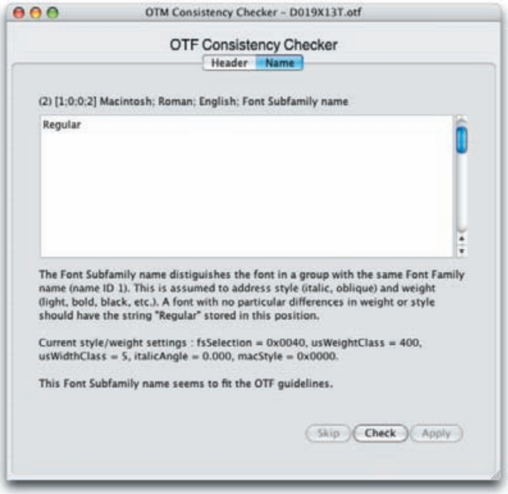
$$\tan -\text{italicAngle} = \text{slopeRun} / \text{slopeRise}$$

The vhea table’s **caretSlopeRise** and **caretSlopeRun** can be checked too if this table is present.

Tip: If your font is a CFF-based OpenType font, also compare with the ‘CFF’ top dictionary ***ItalicAngle****, and possibly with slanting as results from the* ***FontMatrix*** *which is located in* ‘CFF’ top dictionary *as a nested table.*

— Name

The Name section of the Consistency Checker helps inspecting individual name table entries and serves as a built-in documentation of the name table nameIDs, pointing out their correlation with other table's entries. So you may flip through your name table's records and compare your entries with the recommendations.



(2) [1;0;0;2] Macintosh; Roman; English; Font Subfamily name

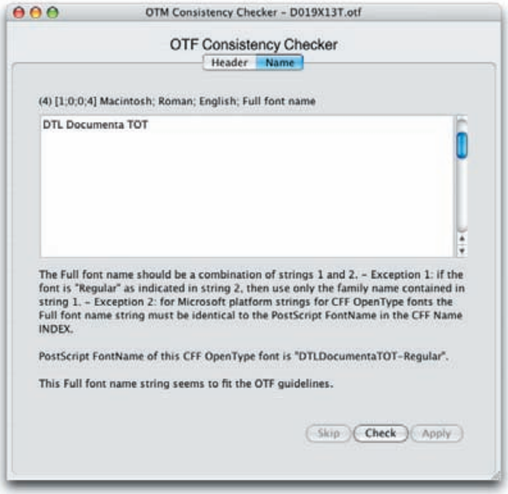
Regular

The Font Subfamily name distiguishes the font in a group with the same Font Family name (name ID 1). This is assumed to address style (italic, oblique) and weight (light, bold, black, etc.). A font with no particular differences in weight or style should have the string "Regular" stored in this position.

Current style/weight settings : fsSelection = 0x0040, usWeightClass = 400, usWidthClass = 5, italicAngle = 0.000, macStyle = 0x0000.

This Font Subfamily name seems to fit the OTF guidelines.

Reviewing the name record with NameID 2 (the Subfamily *name, Macintosh platform).*



(4) [1;0;0;4] Macintosh; Roman; English; Full font name

DTL Documenta TOT

The Full font name should be a combination of strings 1 and 2. – Exception 1: if the font is "Regular" as indicated in string 2, then use only the family name contained in string 1. – Exception 2: for Microsoft platform strings for CFF OpenType fonts the Full font name string must be identical to the PostScript FontName in the CFF Name INDEX.

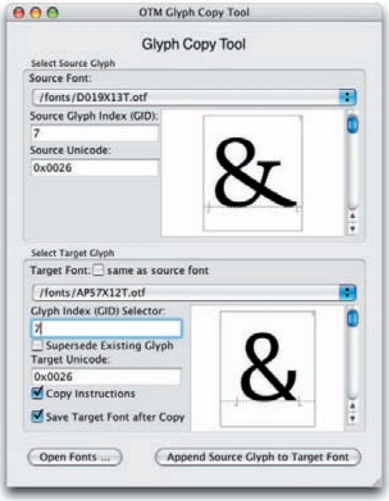
PostScript FontName of this CFF OpenType font is "DTLDocumentaTOT-Regular".

This Full font name string seems to fit the OTF guidelines.

*Reviewing the name record with nameID 4 (*Full *name, Macintosh platform).*

Glyph Copy Tool

This tool makes it easy to copy a glyph from one font to another, or to duplicate a glyph within the same font e.g. to add a same-looking but differently named and reencoded character.



Source Font: /fonts/D019X13T.otf
Source Glyph Index (GID): 7
Source Unicode: 0x0026
Target Font: ☐ same as source font
/fonts/AP57X12T.otf
Glyph Index (GID) Selector: 7
☐ Supersede Existing Glyph
Target Unicode: 0x0026
☑ Copy Instructions
☑ Save Target Font after Copy

The Glyph Copy Tool's dialog.

— Select Source Glyph

Source Font offers all fonts opened in OTMaster in a popup menu. To copy a glyph from a font not opened yet, use **Open Fonts …** to add it to popup list.
Source Glyph Index (GID) is the index of the glyph you want to copy.
Source Unicode is the Unicode codepoint of the glyph you want to copy. So you may determine the source glyph either by its index, its Unicode codepoint, or by using the scrollbar next to the glyph image to flip through all glyphs and choose by appearance.

— Select Target Glyph

Target Font offers all fonts opened in OTMaster in a popup menu.
In case that you intend to merely duplicate a glyph within the same font, activate the checkbox **same as source font**.
Glyph Index (GID) Selector does not have a function as long as you do not intend to supersede (i.e. replace) an existing glyph. Glyphs always are appended as the last glyph of the font.

Note: When copying a glyph, the newly appended glyph's name is derived from its destination glyph index. To give it a friendlier name, you may want to open the Glyph Viewer and change the 'CFF' id or 'post' name.

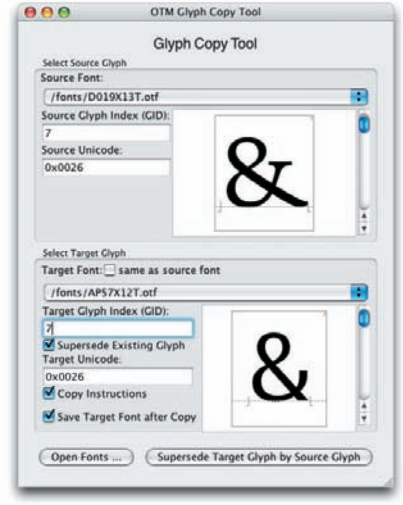
Supersede Existing Glyph – rather than appending the new glyph to the glyph set, activating this option will replace an existing glyph which is defined by the **Target Glyph Index (GID)** right above the checkbox.
Target Unicode is the Unicode codepoint for the new glyph.
Copy Instructions will copy hinting information with CFF-based OpenType fonts.

Note: Hinting instructions are ignored with TT-based fonts.

Save Target Font after Copy will save the target font immediately after appending or replacing a glyph. This is necessary for OTMaster to be able to rasterize the newly appended e.g. in the Font Viewer or Glyph Viewer.

The above image's settings are for appending a font's glyph to another font. As another scenario, the settings in the image below are such that a source glyph will supersede (i.e. replace) the 'same' glyph of another font – here we replace Argo's ampersand by Documenta's:

Tip: For this reason, it is highly recommended that you only edit copies *of fonts with OTMaster, to prevent that you inadvertedly overwrite a font when using* Save *while editing. Either, make a copy of the font file, and edit the copy in OTMaster. Or,* Save As… *a font immediately after opening it, by another name.*



Source Font: /fonts/D019X13T.otf

Source Glyph Index (GID): 7

Source Unicode: 0x0026

Target Font: ☐ same as source font

/fonts/APS7X12T.otf

Target Glyph Index (GID): 7

☑ Supersede Existing Glyph

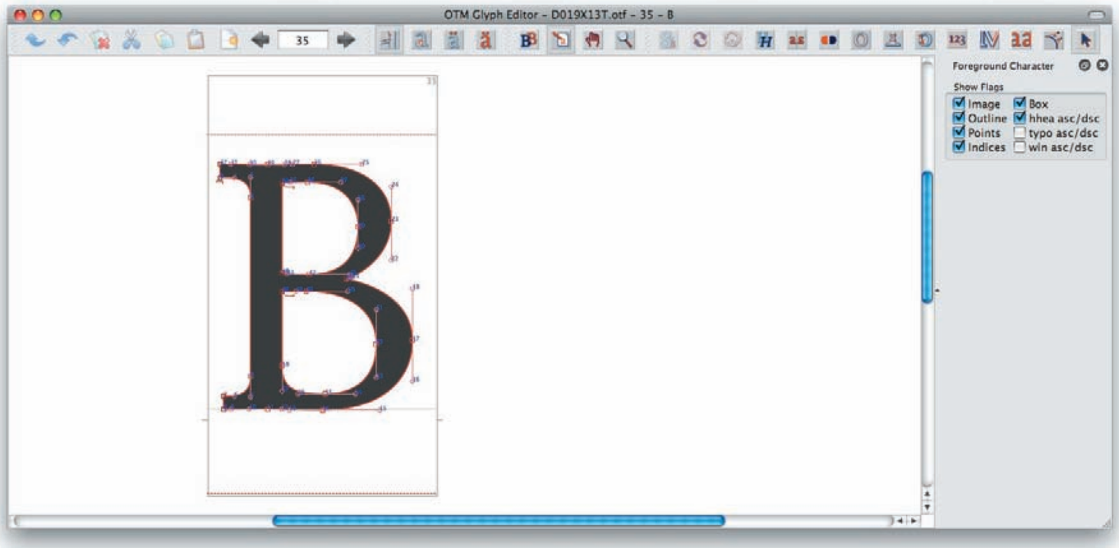
Target Unicode: 0x0026

☑ Copy Instructions

☑ Save Target Font after Copy

Glyph Editor

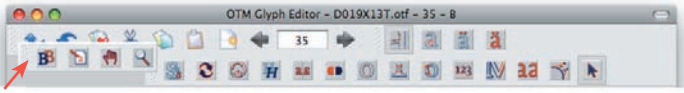
OTMaster is not intended to be a full font editor (font editor in the sense in which FontMaster or its components BezierMaster or IkarusMaster are). But we did not want to miss some basic glyph editing functionality even in a ‘mere’ table editor as OTMaster. What if you find duplicate contours in a glyph of a font ready to be distributed? Go back to the font editor of your choice, open the font’s source data, correct the error, generate the font again, possibly repeat additional production steps with AFDKO or VOLT or at least perform some tricks to smuggle new outline data into the previous font version? With things like these, Glyph Editor will help.



Show Flags

☑ Image	☑ Box
☑ Outline	☑ hhea asc/dsc
☑ Points	☐ typo asc/dsc
☑ Indices	☐ win asc/dsc

The Glyph Editor’s toolbars can be rearranged by simple drag & drop, either in one or more horizontal or vertical columns, or anywhere in the glyph editing area:



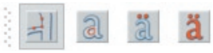
The Glyph Editor’s dialog consists of three areas: a main area for glyph editing, a toolbar area, and the transformation area.

*There are four toolbars: the usual **Edit** functions including a textbox for selecting a glyph by glyph index, different **Selection** modes, **View** (and task) modes, and finally, **Transformation** functions.*

*When opening the Font Editor, the transformation area presents **Show Flags**: like in Font Viewer and Glyph Viewer, you may choose which outline information you would like to see in the glyph editing area. The default settings can be defined in the* ► Preferences.

The Glyph Editor has four toolbars, three of which are reflected in menus, and their functions can be accessed by way of shortcuts too.

SELECTION TOOLBAR



This toolbar, which does not have an equivalent menu, provides different modes for making outline selections:

Select Points

Select individual points by mouse click. Add points to (or remove them from) the selection by holding the ⇧ key and clicking on unselected (or selected) points.

Select Contours

Select individual contours by clicking inside of a contour. Add contours to (or remove them from) the selection by holding the ⇧ key and clicking on unselected (or selected) contours.

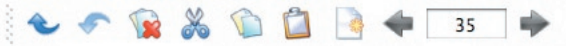
Select Contour Groups

Select contour groups by clicking inside of a contour. Unlike the previous selection mode, Select Contour Groups will select not only the contour you have clicked on but also all contours inside of it. Add contour groups to (or remove them from) the selection by holding the ⇧ key and clicking on unselected (or selected) contours.

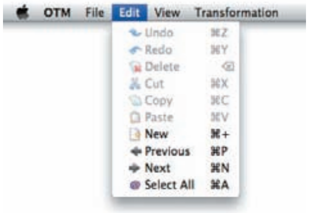
Select Character

Select the entire character by clicking inside of any contour.

EDIT MENU/TOOLBAR



This menu and toolbar holds the basic editing functions like **Copy** and **Paste**, but also **Undo** and **Redo**, and arrows to flip through glyphs.



Undo (⌘+Z) (CTRL+Z)

Un-does previous transformations.

Redo (⌘+Y) (CTRL+Y)

Re-does un-done previous transformations.

Delete (⌫)

Deletes the selected points or contours.

Cut (⌘+X) (CTRL+X)
Cuts the selected points or contours. This will remove them from the glyph and keep them in the clipboard.

Copy (⌘+C) (CTRL+C)
Copies selected points or contours into the clipboard without removing them from the glyph.

Paste (⌘+V) (CTRL+V)
Pastes points or contours from the clipboard into the glyph, or pastes a table entry from the clipboard into the currently selected table.

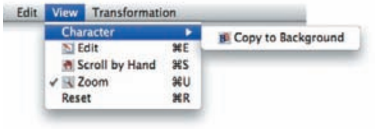
New (⌘++) (CTRL++)
Closes all open fonts. If you have made any changes which have not been saved yet, you will be asked whether to save changed fonts or not.

Previous (⌘+P) (CTRL+P)
Next (⌘+N) (CTRL+N)
Goes to the previous or next glyph. You may also enter the glyph index (GID) into the textbox – and do not forget to confirm with ↵!

Select All (⌘+A) (CTRL+A)
Selects all of the glyph's points or contours.

VIEW MENU/TOOLBAR

This menu and toolbar is essential in that here you choose if you want to edit a glyph, move outlines around in the editing area, zoom in or out, or copy the current glyph into all glyphs' background layer.



Edit (⌘+E) (CTRL+E)
You need to activate this mode whenever you intend to select points or contours, edit points or contours in the **Transformation > Quick Mode** or apply any of the **Transformation** functions.

Scroll by Hand (⌘+S) (CTRL+S)
Click anywhere into the editing area, hold the mouse button down and move the mouse around to scroll the entire editing area.

Zoom (⌘+S) (CTRL+S)
Click into the editing area to zoom in. Hold down the ⇧ key, then click into the editing area to zoom out. The position of the mouse will define the center of the new view. Alternatively, hold down the mouse button and draw a rectangle to zoom into this segment of the editing area.

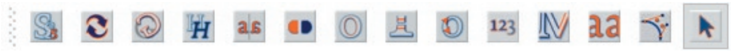
Reset (⌘+R) (CTRL+R)

This will reset to the default glyph size.

Copy to Background

Copies the current glyph into background of all other glyphs. This is useful if you adjust a similar glyph and need something for comparison.

TRANSFORMATION MENU/TOOLBAR



This menu and toolbar offers some functions for transforming glyphs.

Quick Mode

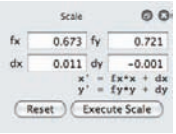
Choose this mode to select and shift selected points or contours.

Shift Smooth

Shifting a point around will interpolate points between the selected point and neighboring extremum points to make sure that curves remain smooth.

Scale Mode

Glyphs, a selection of points or contours can be scaled either with the mouse, or by entering numerical values in the transformation area and then **Execute Scale** to apply scaling or **Reset**.



Scale

fx 0.673 fy 0.721

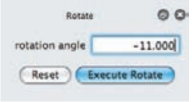
dx 0.011 dy -0.001

x' = fx*x + dx

y' = fy*y + dy

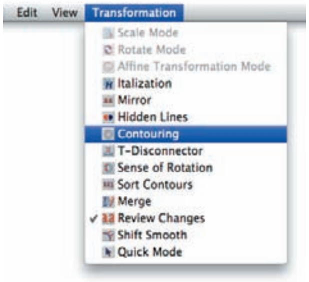
Rotate Mode

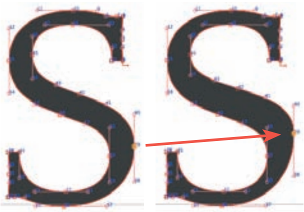
Use the mouse to rotate the glyph in the edit area, or enter a rotation angle in the transformation area (a negative value means clockwise rotation) and then **Execute Rotation** to apply rotation or **Reset**.



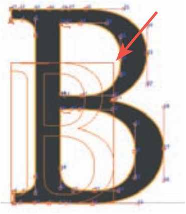
Rotate

rotation angle -11.000





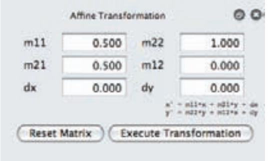
In Shift Smooth mode, surrounding curves (up until neighboring extremum points) will remain smooth when moving a point.



In Scale Mode, click on a corner, hold down the mouse button, and move the mouse to scale the glyph.

Affine Transformation Mode

This will transform the glyph in PostScript font matrix fashion. The equations are given at the bottom right corner of the transformation area:



Affine Transformation

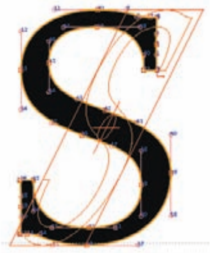
m11 0.500 m22 1.000

m21 0.500 m12 0.000

dx 0.000 dy 0.000

x' = m11*x + m21*y + dx

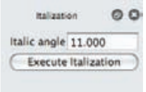
y' = m22*y + m12*x + dy



*An Affine Transformation before applying **Execute Transformation**. Also use the mouse to transform the glyph.*

Italization

This will italizise, or slant, the selected points or contours, either by using the mouse or by entering a value in the transformation area (a positive value will slant to the right side) and confirming with **Execute Italization**:

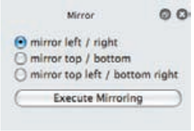


Italization

Italic angle 11.000

Mirror

Mirrors a glyph horizontally (**mirror left / right**), vertically (**mirror top / bottom**) or both ways (**mirror top left / bottom right**):



Mirror

● mirror left / right

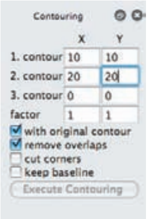
○ mirror top / bottom

○ mirror top left / bottom right

A horizontally mirrored glyph.

Contouring

Use Contouring to add up to three contours around selected contours:



Contouring

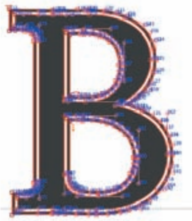
	X	Y
1. contour	10	10
2. contour	20	20
3. contour	0	0
factor	1	1

☑ with original contour

☑ remove overlaps

☐ cut corners

☐ keep baseline



Contouring a glyph with the values specified in the dialog to the left.

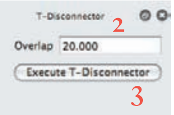
Determine the distance of each additional **contour** from the original outline, independently for **X** and **Y** direction. Distances are given in units relative to upm (the head table's **unitsPerEm**).

With **factor** you may scale the contoured glyph right in one step, again allowing for different values in **X** and **Y** direction. A factor of 1 does not scale the glyph, a factor of 0.5 would scale it down to half the original size, and a factor of 2 would scale it up to double the original size.

As additional options, **with original contour** will keep rather than delete the original contour – deactivating this option and adding a single contour results in bolding of the glyph; **remove overlaps** will remove overlaps resulting from the creation of additional contours; **cut corners** will cut sharp corners that otherwise would touch, or overlap with, other parts of the contour; **keep baseline** will readjust the glyph's position so that the original distance to the baseline is maintained.

T-Disconnector

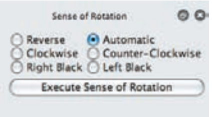
The T-Disconnector will disconnect two parts of a glyph's shapes between any two points, in three simple steps:



Overlap 20.000

Sense of Rotation

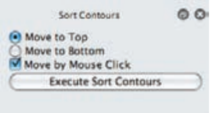
The easiest way to correct contour directions is to select all contours and choose the option **Automatic**. However, you may also select one or more contours (the **Select Contours** mode is recommended for this) and then **Reverse** the current rotation, or define rotation to be **Clockwise** or **Counter-Clockwise**, or by categories **Right Black** or **Left Black**.



○ Reverse ● Automatic
○ Clockwise ○ Counter-Clockwise
○ Right Black ○ Left Black

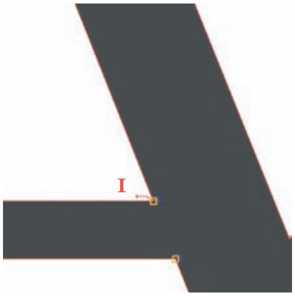
Sort Contours

Select a contour (the **Select Contours** mode is ideal for this), activate **Move to Top** to make it the first one or **Move to Bottom** to make it the last one and then **Execute Sort Contours**. If **Move by Mouse Click** is selected, clicking on a contour will sort contours without need to confirm with **Execute Sort Contours**.

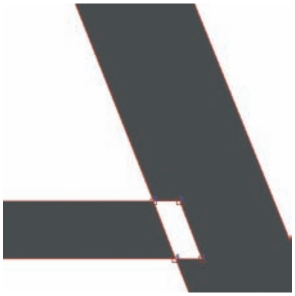


● Move to Top
○ Move to Bottom
☑ Move by Mouse Click

To disconnect e.g. the bar from the rightside diagonal of an uppercase 'A',



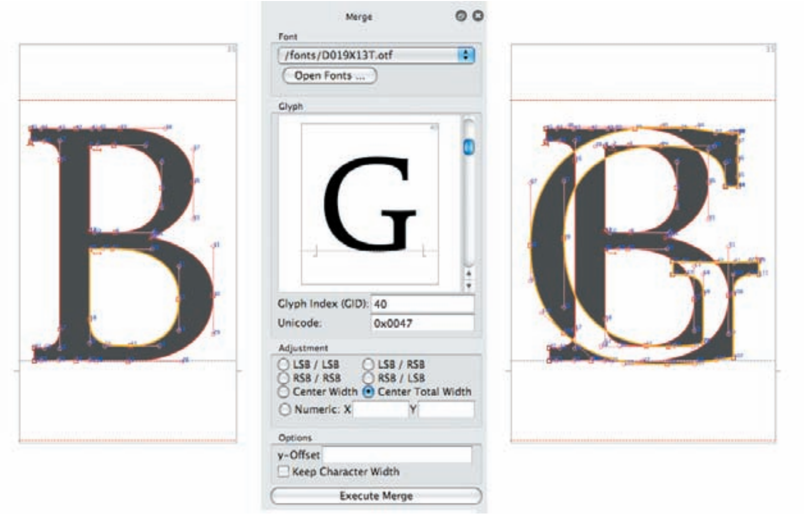
1. select two points at which you want to 'break up' the glyph,
*2. determine the **Overlap** amount in units (relative to the font's upm) in the T-Disconnector transformation area,*
*3. click **Execute T-Disconnector** to apply disconnection:*



Note: In CFF-based OpenType fonts, the outermost contour is expected to be counter-clockwise. In TT-based OpenType fonts, the outermost contour is expected to be clockwise. With both formats though, each contour needs to have the opposite direction than the contour in which it is located.

Merge

To merge one glyph with another one, go to the destination glyph and choose the Merge transformation:



Merge

Font
/fonts/D019X13T.otf
Open Fonts ...

Glyph
Glyph Index (GID): 40
Unicode: 0x0047

Adjustment
○ LSB / LSB ○ LSB / RSB
○ RSB / RSB ○ RSB / LSB
○ Center Width ● Center Total Width
○ Numeric: X Y

Options
y-Offset
☐ Keep Character Width

Execute Merge

Left: the destination glyph. Center: the Merge transformation area. Right: the result of mergin the 'B' with glyph 'G', whereby they are aligned by the ***Center*** *[of]* ***Total Width*** *(including sidebearings).*

Note: You need to remove possible outline overlaps with the ***Hidden Lines*** *transformation.*

— Font

This is the font from which to take a glyph for merging with the destination glyph. You may open a source font with **Open Fonts ...**

— Glyph

This is the glyph to be merged with the destination glyph. You may select it by using the scrollbar to the right of the preview area, or enter either a **Glyph Index (GID)** or **Unicode** codepoint and confirm with ⏎.

*Glyph indices are expected to be integer numbers (0, 1, 123). Unicode codepoints are expected to be hexadecimal numbers, without trailing zeros but preceded by '**0x**' (0x31, 0xB5).*

— Adjustment

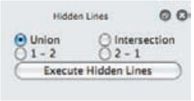
Here you may define how the added glyph shall align with the destination glyph: **LSB/LSB** will position it such that both glyphs' origin points share same coordinates, **RSB/RSB** will align glyphs at the right side, **LSB/RSB** will place the added glyph to the left of the destination glyph, **RSB/LSB** will place the added glyph to the right of the destination glyph, **Center Width** will center-align them (center being calculated from glyphs' outlines, excluding sidebearings), and **Center Total Width** will center-align them (center being calculated from glyphs' total widths, including sidebearings). Or determine the added glyph's position as **X** and **Y** adjustment from the destination glyph's origin.

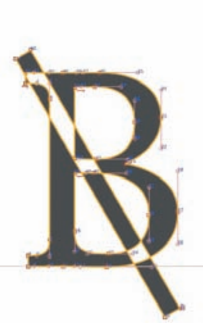
— Options

An additional **Y-Offset**.
Keep Character Width of the destination character.

Hidden Lines

After copying & pasting contours from glyph to glyph or using the Merge transformation, you may end up with contours that overlap. These overlaps are better removed with Hidden Lines:

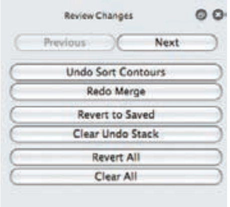


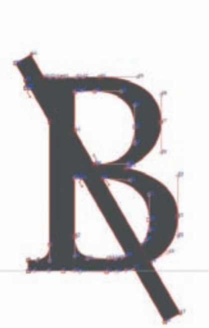

Before removing overlaps with Execute Hidden Lines.

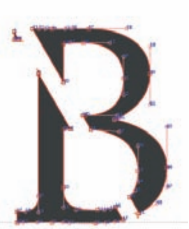
Union will logically add overlapping contours. **Intersection** will keep the overlapping (or intersecting) parts. **1–2** will subtract contour 1 from contour 2. **2–1** will subtract contour 2 from contour 1.

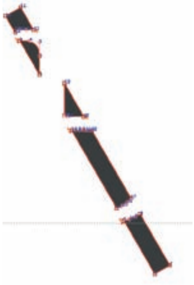
Review Changes

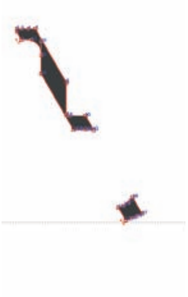
Review Changes presents a history of all glyph changes and allows you to Undo, or Redo, individual steps.




After removing overlaps with ***Union****.*


With ***1–2****: ‘B’ minus ‘backslash’.*


With ***2–1****: ‘backslash’ minus ‘B’.*

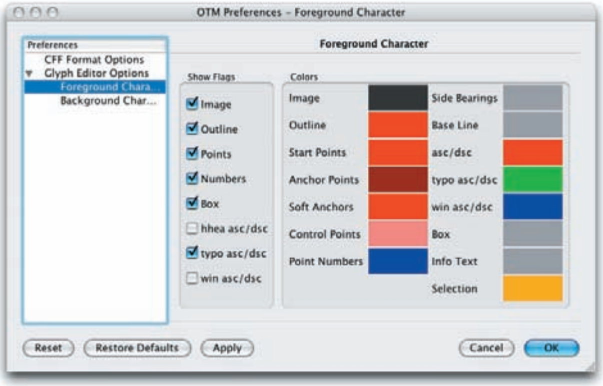

With ***Intersection****.*

Preferences

Currently there are CFF Format Options and Glyph Editor Options.

— Glyph Editor Options

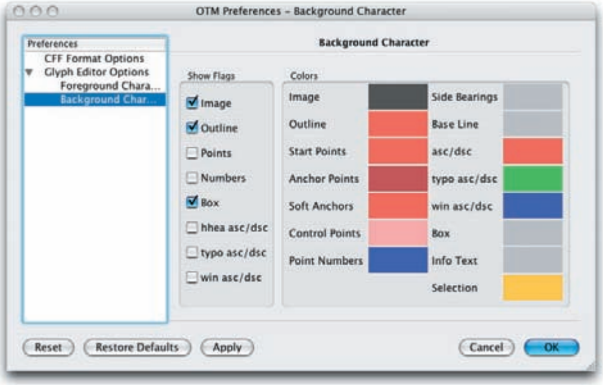
There are two sets of options – for **Foreground Character** and for **Background Character** – both of which offer the same choices:



OTM Preferences – Foreground Character

Foreground Character

Show Flags
- ☑ Image
- ☑ Outline
- ☑ Points
- ☑ Numbers
- ☑ Box
- ☐ hhea asc/dsc
- ☑ typo asc/dsc
- ☐ win asc/dsc



OTM Preferences – Background Character

Background Character

Show Flags
- ☑ Image
- ☑ Outline
- ☐ Points
- ☐ Numbers
- ☑ Box
- ☐ hhea asc/dsc
- ☐ typo asc/dsc
- ☐ win asc/dsc

Checkboxes in the **Show Flags** area define which information the Glyph Editor will visualize by default:
Image shows filled shapes.
Outline adds a colored outline.
Points shows on-curve points.
Indices shows points' index numbers.
Box draws the bounding box around each glyph.
hhea asc/dsc are the hhea table's ascender/descender heights.
typo asc/dsc are the OS/2 table's sTypoAscender/sTypoDescender height.
win asc/dsc are the OS/2 table's usWinAscent/usWinDescent heights.

In the **Colors** area you may adjust the colors of these pieces of information.

Function	Mac OS	Windows
best fitting [values]	**(⌘+B)**	**(CTRL+B)**
Close	**(⌘+W)**	**(CTRL+W)**
Close All	**(⌘+E)**	**(CTRL+E)**
Copy	**(⌘+C)**	**(CTRL+C)**
Cut	**(⌘+X)**	**(CTRL+X)**
Delete	**(⌫)**	**(⌫)**
dec[imal values]	**(⌘+D)**	**(CTRL+D)**
Edit [GE]	**(⌘+E)**	**(CTRL+E)**
Fixup	**(⌘+F)**	**(CTRL+F)**
Grow	**(⌘+G)**	**(CTRL+G)**
hex[adecimal values]	**(⌘+H)**	**(CTRL+H)**
Nested Tables	**(⌘+N)**	**(CTRL+N)**
New [GE]	**(⌘++)**	**(CTRL++)**
Next [GE]	**(⌘+N)**	**(CTRL+N)**
Open	**(⌘+O)**	**(CTRL+O)**
Paste	**(⌘+V)**	**(CTRL+V)**
Preferences	**(⌘+,)**	**(CTRL+,)**
Previous [GE]	**(⌘+P)**	**(CTRL+P)**
Print …	**(⌘+P)**	**(CTRL+P)**
Quit	**(⌘+Q)**	**(CTRL+Q)**
Redo [GE]	**(⌘+Y)**	**(CTRL+Y)**
Reset [GE]	**(⌘+R)**	**(CTRL+R)**
Save	**(⌘+S)**	**(CTRL+S)**
Save All	**(⌘+L)**	**(CTRL+L)**
Save As …	**(⌘+⇧+S)**	**(CTRL+⇧+S)**
Scroll by Hand [GE]	**(⌘+S)**	**(CTRL+S)**
Select All	**(⌘+A)**	**(CTRL+A)**
Sort	**(⌘+R)**	**(CTRL+R)**
Text Dump	**(⌘+T)**	**(CTRL+T)**
Undo [GE]	**(⌘+Z)**	**(CTRL+Z)**
Zoom [GE]	**(⌘+S)**	**(CTRL+S)**

Functions marked with [GE] are available only in the ▶ *Glyph Editor*.

DTL OTMaster *Manual*

Edition 1.0/2009; supports DTL OTMaster version 1.0

Text Karsten Luecke
Design Karsten Luecke
Typeset in DTL Argo, DTL Haarlemmer and DTL Haarlemmer Sans

Thanks to Microsoft for allowing the inclusion of the OpenType 1.5 specs

Dutch Type Library

Zwaenenstede 49
5221 KC '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 and www.fonttools.org

URW++ *Design & Development GmbH*

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@urwpp.de
website www.urwpp.de