_Font_LAB

Type**Tool** 3

BASIC FONT EDITOR FOR FAST AND EASY FONT
CREATION, CONVERSION AND MODIFICATION
USER'S MANUAL FOR WINDOWS

TypeTool® 3

for Windows®

User Manual

TypeTool 3.0 user manual, edition 3.0 [01.03.2007]

Copyright © 1992–2007 by Fontlab Ltd. All rights reserved.

Editors: Sasha Petrov, Adam Twardoch, Ted Harrison, Yuri Yarmola

Cover illustration: Paweł Jońca, pejot.com

This document was created by **Fontlab Ltd** (http://www.fontlab.com/).

Contents

Contents

Introduction

In 1975, at the ATypI conference in Warsaw, Peter Karow from the Hamburg-based company URW introduced Ikarus, the world's first digital type design system that worked with outline fonts.

In 1985, Adobe Systems created PostScript and the Type 1 font format, which both became standards in publishing. In the same year the Texas-based company Altsys created Fontographer — two years before Adobe Illustrator and three years before Freehand, the first Bézier drawing program for Macintosh computers was a font editor! The program became an instant success and greatly contributed to the process of *perestroika* on the type vending market. Until that time, font creation was a domain exclusive to traditional, well-established large font foundries. With Fontographer, the process of creating type was laid into the hands of an average graphic designer. In its first years, this democratization of font making resulted in an outburst of experimental, post-modernist typefaces that are today long forgotten. But the new generation of type designers gradually matured, and so did their tools.

In the early 1990s, Apple introduced the TrueType font format and the Unicode Consortium published the Unicode Standard. Both initiatives laid the foundations for multilingual text processing and were subsequently implemented in Microsoft Windows and Mac OS. In 1993, a group of Russian developers from St. Petersburg led by Yuri Yarmola created FontLab for Windows, which was to become Fontographer's main competitor on the font editor market. In the 1990s, Altsys merged with Macromedia, which released Fontographer 4.1.4 for Mac in 1996. In 1997–98, the developers of FontLab released a series of new products: TypeTool, a basic font editor; ScanFont that allowed designers to quickly convert scanned artwork into fonts; and finally FontLab 3.0, a professional font editor that boasted native Unicode support and sophisticated TrueType hinting.

The turn of the millennium brought about OpenType, a significant initiative that unified PostScript, TrueType and Unicode, and added a sophisticated system of advanced typographic features. In 2001, Fontlab Ltd. released FontLab 4 — the first commercial font editor with OpenType support, followed by AsiaFont Studio, a professional CJK font editor; TransType 2, a universal font converter, and TypeTool 2.

In 2005 — thirty years into Peter Karow's ground-breaking invention — Fontlab Ltd. released FontLab Studio 5, a completely revamped version of the high-end font editor, and acquired Fontographer from Macromedia. The multinational team at Fontlab Ltd. stays at the forefront of digital font technology by bringing quality font software to the typographic community.

The ongoing development of the digital font technology makes it easier for end-users to do text processing, typesetting and layout without sacrificing the typographic quality and logical correctness of the text. But nothing gets lost in Nature: using fonts is getting easier but developing them is more complex. Apart from just drawing letters, a type designer needs to know about encoding, hinting, layout features and various parameters that need to be set inside of a font.

Fontlab Ltd. offers a wide range of outline font editors aimed at different users.

- *TypeTool* (http://www.fontlab.com/typetool/) is a basic font editor for students, hobby typographers and creative professionals who occasionally need to create or customize Type 1, TrueType and OpenType fonts.
- *Fontographer* (http://www.fontlab.com/fontographer/) is a font editor for graphic designers and typographers who would like to create or modify Type 1 and TrueType fonts, using a highly-streamlined, very easy-to-use interface.
- *Fontlab Studio* (http://www.fontlab.com/studio/) is a comprehensive high-end font editors used by font foundries, professional type designers, typographers and graphic design studios, allowing them to create and modify fonts in all major outline font formats, including Type 1, TrueType, Multiple Master and OpenType.
- *AsiaFont Studio* (http://www.fontlab.com/asiafontstudio/) is high-end CJK font editor, based on FontLab Studio but extended with abilities to handle large Japanese, Chinese and Korean fonts.

TypeTool 3.0 is a new version of the basic editor from Fontlab Ltd., allowing the designer to create fonts from start to end. Despite the simple user interface and very affordable price, fonts created in TypeTool are of professional quality.

Major new features of TypeTool 3.0

- Better glyph design: true tangent points, color-customized and streamlined glyph window
- New mask layer editing
- New metrics and kerning editing
- Bitmap background support: import bitmap image or BDF files
- Unicode 4.1 support, new Unicode glyph template images (from Monotype Imaging)
- OpenType fonts import and export
- Better font proofing with new printing modes
- Open and save enhancements: open installed fonts, preview fonts before opening, save all, revert
- Redesigned preferences; save, open and exchange preference profiles
- Better autohinting with Flex Type 1 hints

Other key features of TypeTool

- Outline editor with more than 20 tools and 200-level undo/redo
- Open, edit and generate OpenType PS, TrueType / OpenType TT and PostScript Type 1 formats with up to 65000 glyphs
- Import and export of individual glyphs in EPS format
- Metric and kerning editing module with customizable autospacing and autokerning features
- Import and export of font metrics files in PFM and AFM format
- Automatic Type 1 and TrueType hinting
- Automatic transformation of glyphs
- VectorPaint tools
- Support of two encoding modes and an unlimited number of encoding tables
- Easy-to-use drag/drop-based user interface
- Popup menus and property panels everywhere
- Sample printing of fonts and individual glyphs
- Smooth outline preview

About this Manual

This manual covers the Windows version of TypeTool 3.0.

The following chapters describe all features of TypeTool in full detail. The manual first covers tasks that you need to do first, after installing the application. Later, many of the daily, typical tasks are described. At the end, more specialized topics are discussed.

TypeTool User Interface

This chapter covers the basic definitions of the TypeTool user interface and its customization and gives a short description of all the editing windows and panels. All TypeTool options are discussed there.

Editing Fonts

Fonts consist of glyphs — graphic objects that depict letters, digits, or symbols of various kind. Each glyph needs to be encoded, so that pressing a key on the keyboard produces the intended symbol on the screen. This chapter explains how to manage the font's glyph repertoire, set and change the glyph encoding, copy and rearrange glyphs within a font and between fonts, select glyphs for editing, and edit font info fields.

The Font Header

Fonts are not just collections of glyphs. Each font has some central "header" information such as menu names, copyright information, linespacing information, style-linking within a family. This chapter explains the most important aspects of the font header and the TypeTool tools for managing them.

Printing and Proofing Fonts

Viewing single glyphs on the screen in a close zoom rarely gives you the appropriate impression on how the font will look in printed text. This chapter provides a detailed description of how to print from the Font, Glyph and Metrics windows. Other font proofing methods are also described in this chapter.

Generating Fonts

When the design is finished, the font needs to be produced in a final format that can be installed on a user's machine. This chapter explains how to generate fonts in different formats and how the generation can be controlled using a variety of options.

The Glyph Window

Type design is much, but more than anything it is about drawing shapes. This chapter concentrates on the tools used in the glyph design process.

Editing Metrics

Glyphs rarely appear as isolated images; more often, they are shown in running text, so the distances between them need to be carefully edited. TypeTool lets you define the glyph advance widths, sidebearings, and kerning manually or automatically, and this chapter discusses these aspects in detail.

Actions

Scaling, rotating, flipping or cleaning up outlines; increasing advance widths, equalizing sidebearings; removing hints, applying effects — actions can help you to speed up your work. This chapter gives detailed descriptions of all actions and their usage in TypeTool.

OpenType Fonts

This chapter explains how to open and generate OpenType fonts with TypeTool.

System Requirements

The Windows version of TypeTool requires one of the following hardware and software configurations:

A PC computer capable of running one of the following versions of Windows: Windows 98, Windows ME, Windows NT 4.0, Windows 2000 or Windows XP with one of these operating systems installed.

At least 10Mb of free space on the hard disk drive and at least 64 MB RAM. TypeTool will start on 32 MB RAM but you will need more RAM to open bigger fonts.

TypeTool User Interface

Before talking about fonts and the TypeTool font-editing features, let's spend some time learning the TypeTool user interface. For the most part, it is similar to the user interface of your operating system. If you know how to navigate in Windows and in Microsoft Office, you will feel comfortable with TypeTool. Some other parts of the user interface are unique — that is what we will focus on.

The user interface of TypeTool is highly customizable. This chapter discusses how you can custom-tailor the TypeTool interface so it fits your needs best. Note that throughout the book, we will refer to menu commands, toolbar buttons and keyboard shortcuts as they appear in the default user interface settings of TypeTool, so keep this in mind if you have customized their location or appearance.

Basic Terms

First things first, let us define some terms that are critical to understanding TypeTool and fonts in general.

Character

The minimal, atomic unit of writing with a clearly defined identity — a part of the alphabet, a letter, a digit, an ideogram, a symbol.

Any image that can be recognized as having the same meaning represents the same character:

AA ⓐ A A

All the images above represent the character "A".

Please note that sometimes, identical images represent different characters:

A Latin "A"

A Cyrillic "A"

A Greek "Alpha"

Characters are abstract beings without a particular, strictly defined image. Computers store characters in their memory using numerical *codes*. A text file contains sequences of such codes that represent strings of characters.

Glyph

The basic, atomic element of a font, the particular image that is being shown on screen or printed. The glyph repertoire of a font is a collection of all glyphs contained in this particular font. Typically, one glyph is a graphical representation of one character. However, the same font can include several glyphs that are different graphical representations of one and the same character:

tt,t't

Also, one glyph can represent several characters, for example in a ligature.

Characters are the abstract, conceptual components of a text, while glyphs are the particular, visual components of a text fixed in some form.

In addition to the visual appearance (the glyph image), a glyph also has some digital representation. A glyph can be represented by a bitmap, capable of reproducing the glyph image only in one specific size. More commonly, a glyph consists of outlines that are scalable so that they can reproduce the glyph image in any size.

Font

A typeface is a particular artwork of an alphabet, a set of glyphs that are designed with a common graphic idea in mind.

A font is a digital file (or a few files) that holds a digital representation of a typeface. A font contains an organized collection of glyphs along with some additional information that defines the spatial relations between the glyphs (metrics and kerning) as well as some central parameters such as names, copyright information, linespacing values etc. (font header).

Encoding

When the user presses a key (or a combination of keys) on a keyboard, a numerical code is stored in the computer's memory. This code represents a character, an abstract unit of writing. A series of such codes forms a string of text.

Every keystroke stands for a different character, so every character uses a different numerical code (also called codepoint). Operating systems and applications need to know which number represents what character — otherwise a spelling checker couldn't recognize words that you're typing. So each operating system and each application uses a list that maps characters to numerical codes. This mapping is called "text encoding". In the past, different computers used different text encoding standards (so-called codepages), so that the letter "Ä" was stored as number 142 in one program and as number 128 in another program. Modern operating systems and applications encode text using the international Unicode Standard, which assigns a unique numerical codepoint to every character used in writing by humankind. In Unicode, "Ä" always uses the code U+00C4 (which is hexadecimal for 196).

Whenever the text is printed or displayed on the screen, the computer looks inside the font file and finds out which character codes correspond to which glyphs, so that the series of abstract character codes can be visualized using some specific images of letters, digits and other symbols. Every font includes a mapping of character codes to glyphs — this mapping is called the "font encoding" (sometimes, "encoding vector", or, in this manual, just "encoding"). These days, most font formats use the Unicode Standard as basis for their font encoding, but it is possible to produce fonts that are encoded using of the older codepages. It is even possible to have several different encoding vectors in one font, so that both old and modern applications can work with it.

Font Family

A font family is a collection of fonts representing typeface designs that share the same design idea but differ in width, inclination (upright or italic), stroke thickness (weight), stroke endings (serif or sanserif) or in some other stylistic aspects.

For example, "Times Bold Italic" is a typeface that belongs to the "Times" family. "timesbi.ttf", "Times-BolIta.otf" and "tmbi_____.pfb" are all different fonts representing the same typeface in different formats.

A font family usually contains from one to a few dozen typefaces.

Glyph name

Each glyph in a font contains the digital representation of the glyph image (in form of outlines or bitmaps). In addition, each glyph is uniquely identified by its name. Glyph names must follow certain conventions:

1. Only plain English letters (uppercase A-Z or lowercase a-z), European digits (0-9) as well as the special characters "." (period) and "_" (underscore) are permitted in glyph names.

2. Spaces are not permitted in glyph names!

3. Glyph names must not start with a digit.

4. Glyph names must not start with a period except the glyph name ".notdef". ".notdef" is a special glyph that is displayed by the operating system if the font does not include a codepoint that the application is attempting to display. Usually, ".notdef" has a form of a rectangle, a crossed rectangle, or a rectangle with a question mark inside.

Additional glyph naming conventions are discussed in the ***Glyph Naming and Character Encoding*** (on page 93) chapter.

Menu

When we refer to menu items in the main TypeTool menu, we will use the following notation:

[top menu item] > [sub-item]

For example:

Edit > Copy means: click on the word **Edit** on the menu bar and select the **Copy** command from the menu:
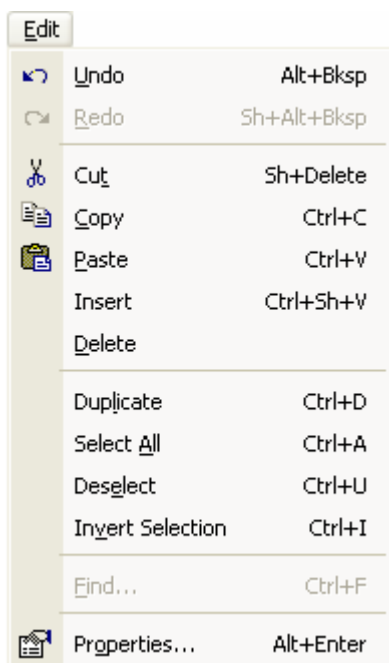
Folders and Paths

Recent applications from Fontlab Ltd. use a new folder structure for storing their data files such as encoding or codepage definitions, glyph generation recipes, text samples for metrics and kerning, mapping tables etc. TypeTool 3.0 looks for data files in four different folders.

Shared default data folder

typically, C:\Program Files\Common Files\FontLab

This folder holds files that are commonly used by all recent Fontlab Ltd. applications: FontLab Studio 5, TypeTool 3, TransType SE/Pro, FogLamp, SigMaker 2, BitFonter 3, with more to come. In each respective subfolder, codepage definitions, encoding definitions, glyph-to-Unicode mapping files and some special data files are stored. Only Fontlab Ltd. applications and applications from registered Fontlab Ltd. developer partners should place their files there. This is to rule out conflicts between the user's customized files and default files.

Shared user data folder

typically
C:\Documents and Settings\Your Username\My Documents\FontLab\Shared

This folder has exactly the same structure as the folder discussed above and can store any files customized by the user. Any file placed in the respective location within that folder will override the corresponding file placed in the shared Fontlab Ltd. Please put your customized files in this folder. The location of the folder can be modified in Tools > Options > General Options > Folders and paths:


```
User data files for all FontLab products:
C:\Documents and Settings\Alex\My Documents\FontLab\Shared\
```
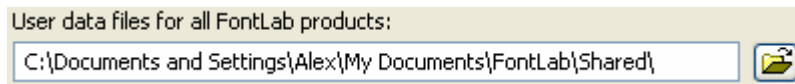
Application default data folder

typically, C:\Program Files\FontLab\TypeTool3

This folder holds files that are only used by TypeTool. In each respective subfolder, metrics, kerning and other text strings, additional encodings as well as samples are stored. Only Fontlab Ltd. applications and applications from registered Fontlab Ltd. developer partners should place their files there. This is to rule out conflicts between the user's customized files and default files.

Application user data folder

typically,
C:\Documents and Settings\Your Username\My Documents\FontLab\TypeTool3

This folder has exactly the same structure as the folder discussed above and can store any files customized by the user. Any file placed in the respective location within that folder will override the corresponding file placed in the shared Fontlab Ltd. Please put your customized files in this folder. The location of the folder can be modified in Tools > Options > General Options > Folders and paths:

TypeTool 3 user data files:
C:\Documents and Settings\Alex\My Documents\FontLab\TypeTool3\
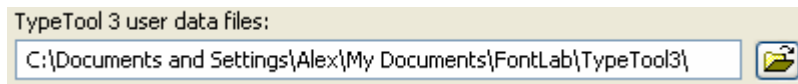
When we refer to one of the folders, we will use the following syntax:

[main folder]/[subfolder name]

Where [main folder] can be one of the following: [Shared default data folder], [Shared user data folder], [Application default data folder], [Application user data folder], and [subfolder name] is the name of the particular subfolder within that folder.

For reasons of brevity, we will sometimes write:

[Shared] which will mean either [Shared default data folder] or [Shared user data folder]

[Application] which will mean either [Application default data folder] or [Application user data folder]

This means that a particular file can be stored in either of the two locations (default or user). Remember that user locations always override default locations.

Mouse

Click on the mouse on some object	Position the mouse cursor on the object and click the left mouse button
Right-click on some object	Position the cursor on the object and click the right mouse button
Ctrl-click on something	Position the cursor over "something", hold down the **CTRL** key on the keyboard and click the left mouse button.
Drag some object	Position the cursor on the object, press and hold down the left mouse button and move the mouse to move the object, without releasing the mouse button while moving. Release the mouse button when you're done.

Context Menu

Most windows and panels in TypeTool have context menus associated with them. These menus contain the most useful or most frequently used operations that the user may have to perform in a particular situation. To open the context menu, right-click (see the section *Mouse* (on page 28)) on an empty area in the window or panel, or on a particular object (e.g. a glyph or a selection or a node). Remember that the context menu will change depending on the context, i.e. right-clicking on a particular object will often display a different context menu than right-clicking on an empty area of a window.

Getting Started

When you run TypeTool for the first time (to run TypeTool double-click on

its icon) you will see a welcome screen for a few seconds and then the
TypeTool window:
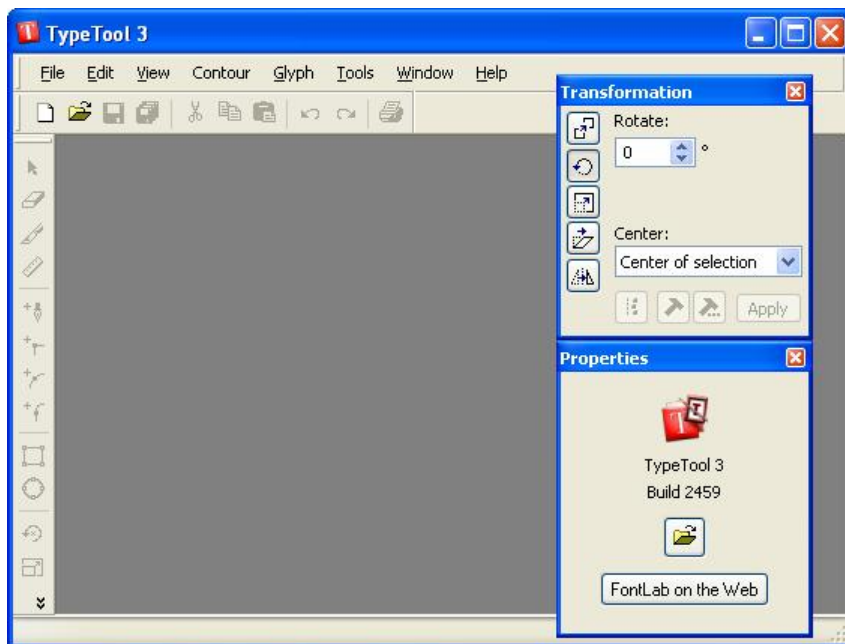
Like almost all Windows programs TypeTool has a menu, a few toolbars
and a status bar at the bottom.

You can position the TypeTool menu at any place on the screen. The usual
location is at the top of the screen, but if you want to put it somewhere else,
drag it there:
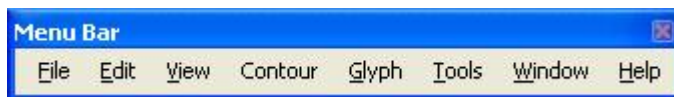
Any toolbar can be repositioned in the same way — you can leave them at
the top or drag them anywhere.

29

You can easily choose which toolbars you want to see: use the **Toolbars** command in the **View** menu or simply click the right mouse button on a menu, toolbar, or toolbar docking panel and you will get exactly the same menu:

✓	Status <u>B</u>ar
✓	Standard
▦	Tools
✓	P<u>a</u>int
✓	<u>C</u>ontour
	<u>M</u>etrics Tools
	M<u>e</u>trics Commands
	<u>C</u>ustomize…
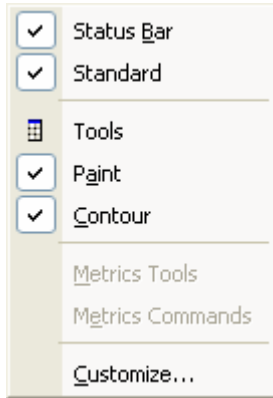
Following is a list of common toolbars with a few comments about each:

Status Bar	Status bar at the bottom of the window
Standard	Contains basic commands like file open and save, copy/paste, print and help
Show Layers	Controls the appearance of basic *Editing layers*
Tools	Probably the most important toolbar — gives access to editing tools that you will use to work on glyph shapes

You may notice a few *italic* terms. We will describe them later. Specifically, *Editing layers* in the "***Glyph Window*** (on page 139)" chapter.

OK, we are almost ready to open a sample font, but before we do let's talk about *customization* of the TypeTool user interface.

Customizing TypeTool's User Interface

As you may infer from the title of this section most of the TypeTool user interface (which means menus, toolbars and keyboard shortcuts) is customizable. We believe our default interface is the easiest to use, but if for some reason you do not like it, you are free to make any changes you want. If you do not want to change anything in the TypeTool user interface, you can *fast forward* to the next section.

The general idea of customization is simple: there is a long list of commands that you can use and three kinds of controls: menus, toolbars and keyboard shortcuts. Through customization you can assign any command to a menu item, button on a toolbar or combination of keys pressed on a keyboard. In addition you can organize commands in context menus or toolbars.

Most of the customization commands are concentrated in the Customize panel that you can open with the **Customize** command from the **Tools** menu or the same command located in the context menu which appears if you right-click on a menu, toolbar or toolbar dock area:

The **Customize** dialog box consists of several pages:

Commands	List of all the available commands grouped into several categories
Toolbars	Customization of toolbars. There is an option to create new toolbars.
Tools	On this page you can "connect" an external program to a menu item in TypeTool's Tools menu
Keyboard	Customization of keyboard shortcuts
Menu	Customization of menus

While the **Customize** dialog box is open all interface elements are in "editable" mode, so you can simply drag-drop buttons and menu items between different toolbars. You can also customize the appearance of menu items and toolbar buttons.

Customizing Toolbars

To move a button within a toolbar, click the left mouse button on it; drag it to the new location and drop it. If you drag the button slightly further to the right, a separator bar will be added between it and the previous button:

To move a button to another toolbar, drag-drop it there. To copy a button, hold down the CTRL key while you drag the button.

To remove a button from a toolbar, drag it out of the toolbar:
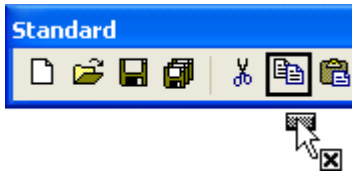
In TypeTool there is very little difference between a menu and a toolbar, so you can re-arrange, copy or remove menu items just like you did with toolbar buttons:
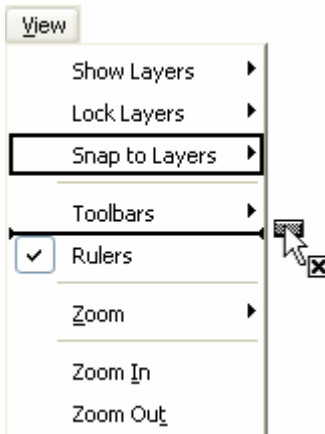
You can also drag a menu item onto a toolbar to add a toolbar button. Hold the CTRL key to copy the item.

To get access to all the TypeTool commands, open the **Commands** page in the Customize dialog box:
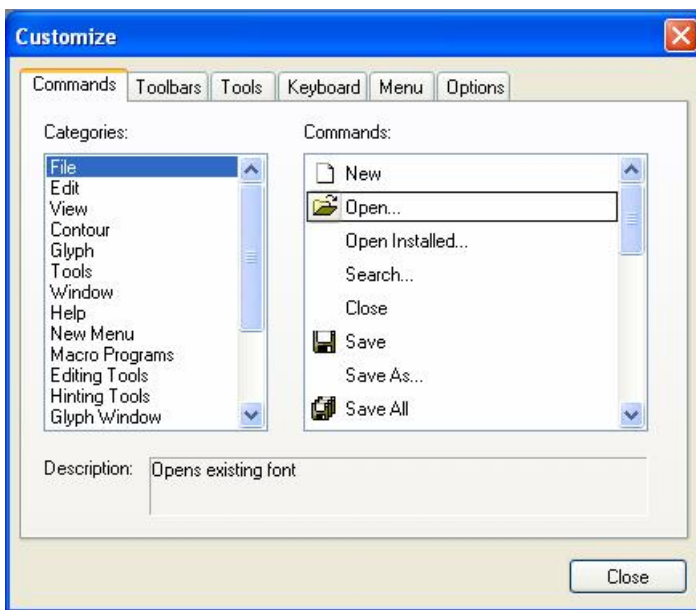
In the left list select a group of commands and use the list of commands in the right list as a source of menu items and toolbar buttons: drag the commands from there.

Customizing Menus

If you want to create a new context menu, select the **New Menu** group in the left menu and drag it onto the main menu bar or any of the toolbars. A new menu appears and you can start adding commands to it using the drag-drop technique described above.

With the Customize dialog not only can you customize the main menu, but also most of the context menus which appear when you right-click on TypeTool windows. Open the **Menu** page in the Customize dialog box and choose a context menu in the right combo box:
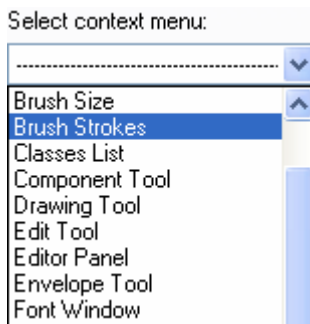
A menu appears on screen and you can customize it by dragging commands from the toolbar, other menus or the list of the commands on the **Commands** page.

To reset changes you have made in menus, use the **Reset** buttons on the **Menu** page of the Customize dialog box. Use the left **Reset** command to reset changes in the main menu and the right **Reset** button to reset changes in the context menus.

Customizing Individual Items

You can customize the appearance of any menu item or toolbar button. The following appearances are available for most items:

Image

Text   New Font

Image and Text   New Font

To change the appearance of the menu item or toolbar button position the mouse cursor on the button and click the right mouse button. Select the new appearance method in the context menu:

> Reset to Default
> Copy Button Image
> Delete
>
> Button Appearance...
> ✓ Image
> Text
> Image and Text
>
> Start Group

Most commands in TypeTool have pre-designed images, but you can easily create your own images for any toolbar button or menu command. To do so, select the **Button Appearance** command in the button's context menu:

Button Appearance...

You will see a dialog box where you can choose the appearance method and, if it includes an image, choose the image that appears on the button or at the left of the menu item:
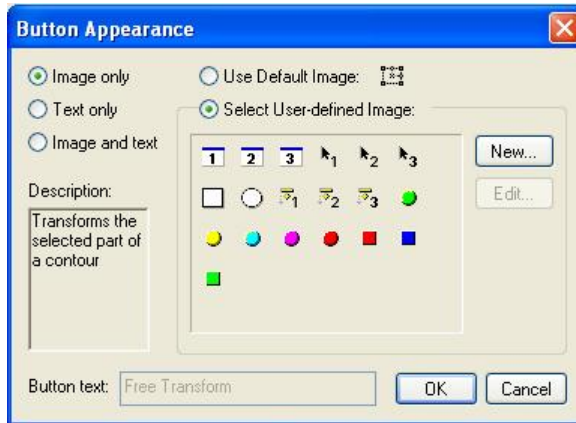
Choose a User-defined image; click on **New** to create a new image; or **Edit** to edit one of the User images. If you decide to change the image, use the included image editor to change it:
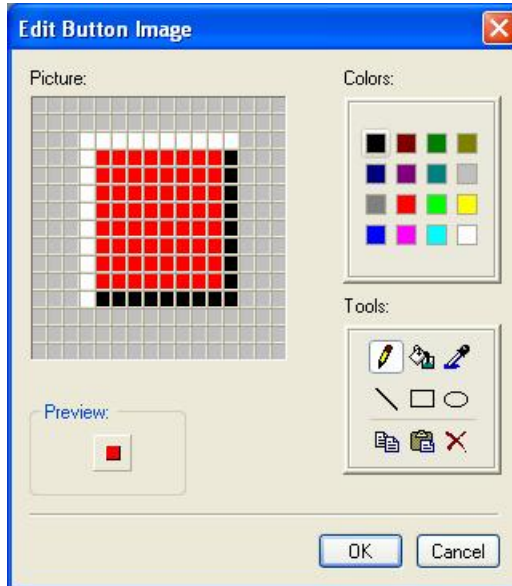
Use one of the **Tools** to edit the enlarged image and choose a color in the **Colors** area. Click on the **OK** button when you are ready.

Converting a Menu to a Toolbar

In TypeTool some menus can be converted to toolbars. If you open a context menu and can see a tiny blue caption in the top area of it, you can drag it to any place on screen and it becomes a toolbar:
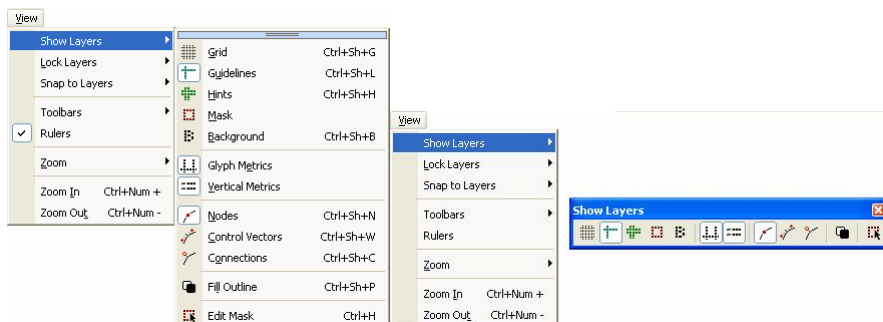
Not all menus have this feature, but you may find it really useful. OK, that is almost all about customizing toolbars and menus. A few more things:

To reset changes you have made in toolbars, use the **Reset All** button on the **Toolbars** page in the Customize dialog box:
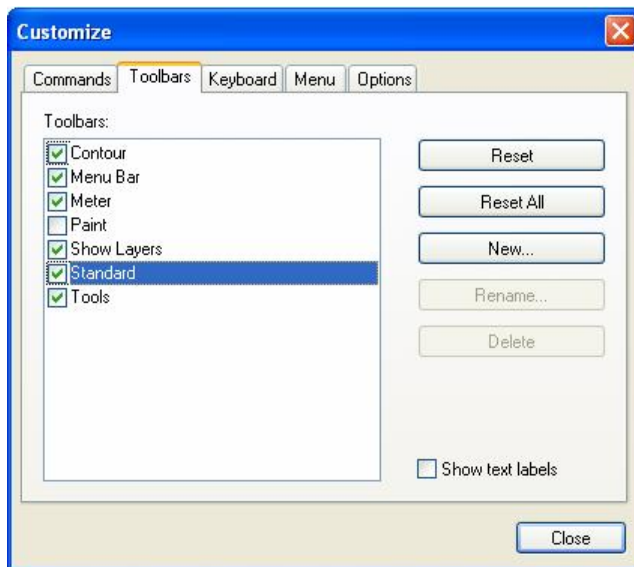
Use the **New** command on the same page to create a new toolbar. After doing this, add commands to it by the drag-drop method described earlier in this section.

37

Customization of the Keyboard

Open the **Keyboard** page of the Customize dialog box:
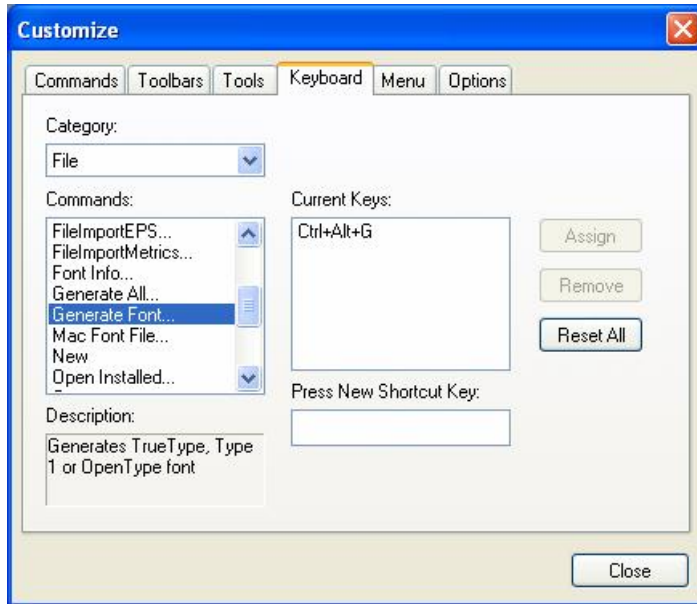
In the left area of the page you can select the command, which you want to customize. Choose the commands category in the top list and the command itself in the list below.

On the right part of the page you will see the list of keyboard shortcuts currently defined for that command:
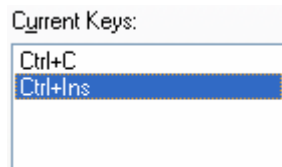
The **Remove** command at the right of the list allows removal of one of the existing shortcuts.

To define a new keyboard shortcut, select a command and position the cursor on the editing field below the **Press New Shortcut Key** label:

Press <u>N</u>ew Shortcut Key:

When the caret is in position, press the combination of keys that you want to assign. A description of that combination will appear in the editing field and you can click on the **Assign** button to assign that combination to the currently selected command.

Press the **Reset All** button to reset all changes back to TypeTool defaults.

There is a **Keyboard shortcuts** command in the **Help** menu. Use it after you finish customizing the user interface to see or print a list of all the shortcuts:

Command	Keys	Description
FileClose		Closes current font
FileExit		Quits the application
FileExportBackgroun...		Saves bitmap background into an image file
FileExportEPS...		Saves current glyph in EPS (Adobe Illustrator)...
FileExportMetrics...		Saves the current metrics information in AFM ...
FileFontInfo...	Ctrl+Alt+F	Edits font header fields
FileGenerateAll...		
FileGenerateFont...	Ctrl+Alt+G	Generates TrueType, Type 1 or OpenType font
FileImportBackgroun...		Opens bitmap background from an image file
FileImportBitmapFont...		Import bitmap font file
FileImportEPS...		Opens EPS (Adobe Illustrator) file
FileImportMacFontFil...		
FileImportMetrics...		Opens a metrics file
FileNew	Ctrl+N	Creates a new font
FileOpen...	Ctrl+O	Opens existing font

Category: File   Show Accelerator for: Default

Help Keyboard

Links to External Programs

Use the **Tools** page of the Customize dialog box to assign Windows programs to menu items in TypeTool's **Tools** menu:
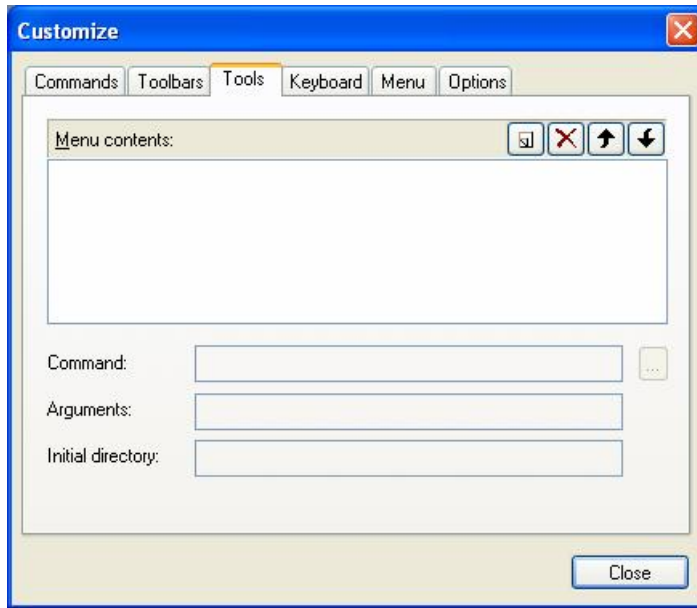
There is a list of the assigned programs in the middle of the page and it is empty by default. Click on this button: ⏺ to add a link.

Enter the name of the program as it will appear in the menu:

ScanFont

Then use the **...** button at the right of the **Command:** editing field to locate the program to run:

...

Editing fields at the bottom of the **Command** field allow you to define arguments for the program you want to run from TypeTool's menu.

Use the ✕ button at the top of the **Tools list** to remove the reference to the program and the ⬆ and ⬇ buttons to change the order of the commands.

You may use special parameters to run external programs with the currently opened font as an argument. When TypeTool recognizes this argument, it will replace it with the file name of the currently active font or with some other parameters.

Suppose that current font was last saved into file named "c:/fonts/sample.vfb".

Special arguments are:

%p	Full file name of the current font [c:/fonts/sample.vfb]
%f	Name of the file with extension [sample.vfb]
%n	File name only [sample]
%x	File name extension [vfb]
%d	Path where file was saved [c:/fonts/]
%a	Path to TypeTool installation [usually it is "/program files/FontLab/TypeTool3"]

Now you know everything about the customization of menus, toolbars and the keyboard, so you can click on the **Close** button at the bottom of the Customize dialog box to exit the customization mode.

➘ Important note: in the following manual we will describe all commands, buttons and keyboard shortcuts as they come with TypeTool, without any customizations. If you changed the interface but want to follow the manual, reset all changes with the **Reset** buttons on the **Toolbars**, **Keyboard** and **Menu** pages of the Customize dialog box.

Faster Method to Customize Commands

You can customize toolbars and menus without opening the Customize dialog box by holding down the **ALT** key on the keyboard and dragging buttons between toolbars or toolbars and menus.

TypeTool Windows

There are three kinds of windows in TypeTool:

Font Window	Gives you the overview of the glyph repertoire of your font and allows you to control the encoding
Glyph Window	Allows you to draw and modify the design of your glyphs
Metrics Window	Allows you to set and modify the distances between the glyphs: metrics and kerning

This chapter provides some basic information about the three main windows. Please refer to the "***Editing Fonts*** (on page 79)", "***Glyph Window*** (on page 139)" and "***Editing Metrics*** (on page 257)" chapters for more detailed information about these windows.

Font Window

To demonstrate the TypeTool Windows, let's create a new font. Use the **New** command in the **File** menu or click on this button ⬜ on the Standard toolbar.

You will see the Font window:
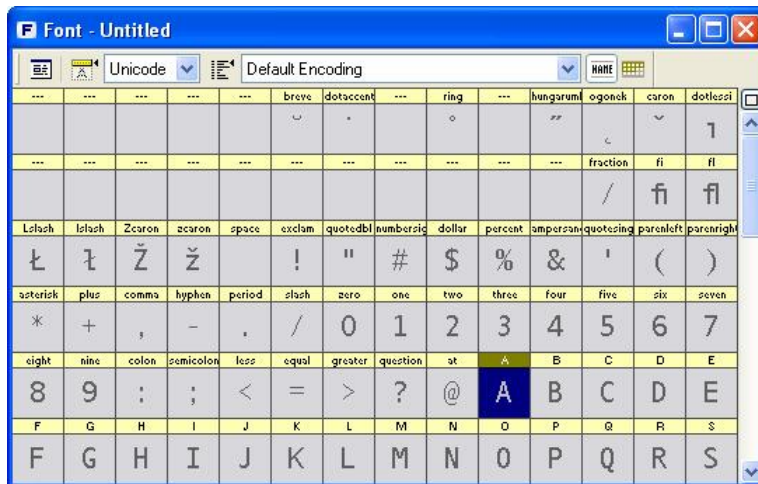
As you can see, this window has a command bar with a few buttons and options and a big table of cells that represent glyphs contained in the font. Each cell has a caption that contains glyph identification information: name, Unicode codepoint or some other data:

The cells can also contain little icons that indicate some glyph properties, but more about that later.

43

There are no glyphs in the font that we just created, but the Font window shows some images in the glyph cells. These are glyph template images that show which character does the specific glyph cell represent. These glyph template images are very simplified images of Unicode characters that the glyphs represent. You should use the glyph template images for orientation, but not necessarily as a definitive source of information about the design of that particular glyph. TypeTool has templates for thousands of glyphs, so you will usually know where to place new glyphs.

We will discuss navigation in the Font window later, in the "***Editing Fonts*** (on page 79)" chapter, so let's talk about the Font window command bar, which is located either at the bottom or at the top of the window:
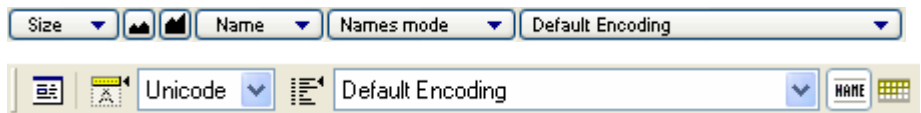
You can switch between the top and the bottom location of the command bar by clicking on this button ⬚ in the top-right corner of the Font Window.

When the command bar is in the top position, it can be detached and you can drag it to the bottom area or leave it floating around:
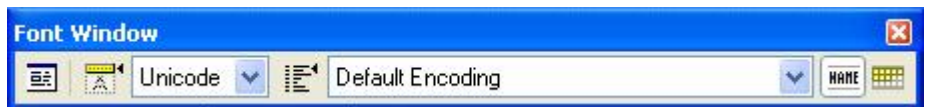
The first button on the command bar (when it is in the top position) 🔳 invokes the **File > Font Info** command, which is described later in this manual.

Next, there is a combo box that allows you to change the information that appears in the glyph cell captions:

Next is a combo box that allows you to change the encoding table of the current font:
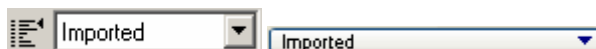
We will talk about encodings later, but you could choose a couple different ones from the combo box and see how the Font window changes.

At the right of the encoding list there are two buttons that allow a choice of encoding modes. Any glyph in the font may be identified by a name or Unicode codepoint; a detailed description of this follows.

Two buttons in the command bar in the top position allow you to choose one of two modes: Names or Codepages.

In the bottom position, there is a combo box that you can use to choose the mode:

That is all about the Font window for now so let's open the Glyph window.

Glyph Window

To open a Glyph window for editing individual glyphs you need to create one. Remember that we started with a new font that does not have any glyphs. To create a glyph, double-click on any cell in the Font window. The gray cell, which indicated that there was no glyph defined ("empty glyph cell") has been replaced by a white cell, which represents a glyph that is defined, but contains no image ("blank glyph"). When you draw or paste something into it, the white cell will show a small image of the glyph.

After a blank glyph is created, we are ready to open the Glyph window. Select the glyph cell (click on it with the left mouse button) and double-click on it to open the Glyph window. It will immediately appear on screen:
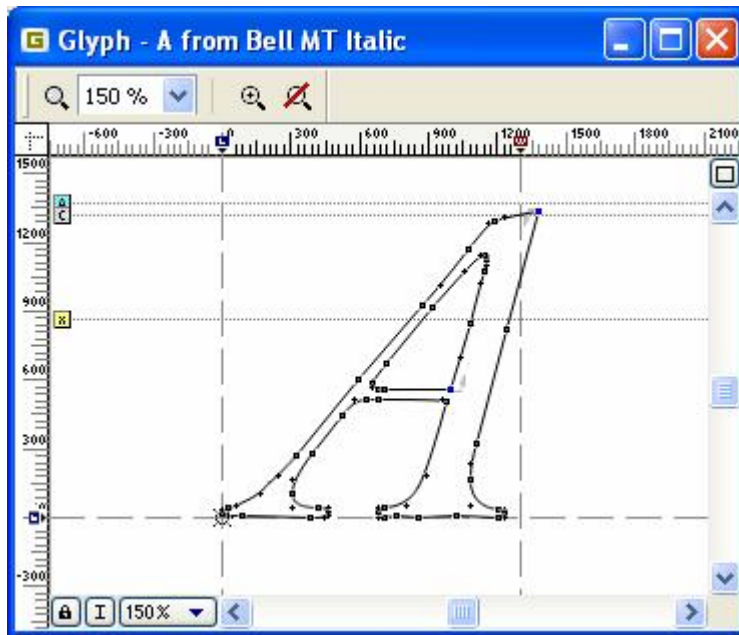
Instead of double-clicking on, you can also use several other methods to open the Glyph window:

1. Right-click on the glyph cell and select the **Open Glyph Window** command in the context menu.

2. Select the glyph and choose **New Glyph Window** in the **Window** menu.

3. And finally, select the glyph cell and press the **ENTER** key on the keyboard.

If you have more than one glyph in your font (which is normal when you open an existing font) and have a glyph window already open when you double-click on another glyph in the Font window (or use any other method of opening a glyph window except the **New Glyph Window** command) a new glyph will appear in the original glyph window. If you need to open many glyph windows simultaneously hold down the **CTRL** key when you double-click on the new glyph cell or otherwise open a new Glyph window.

You may have as many open glyph windows as you want, close those you do not need so as not have all your workspace covered with glyph windows.

Glyph Window Contents

All windows in TypeTool have a similar layout: control panel on the top and main area covering most of the window. The glyph window is no exception: the top-docked control area (which, of course, can be docked to the bottom location also) contains zoom selection tools: a combo box, and a few toolbar buttons:
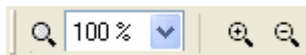
To get more screen space for the editing field you may hide the zoom toolbar if you click on this button in the top-right area of the glyph window:

The main area of the window has scroll bars to change the view of the glyph, and vertical and horizontal ruler bars.

47

You can switch the ruler bars on and off with the **Rulers** option in the **View** menu. A quicker way is to right-click on the ruler and choose the option in the context menu:

At the bottom-left corner of the windows you will find two buttons, Lock and Meter:

The **Lock** button controls quick access to the font glyph — when it is in the "unlocked" state you can use the keyboard to directly access the glyphs. I.e. when you press a key the corresponding glyph will automatically open in the glyph window.

The **Meter** button controls the appearance of the *Meter panel*, which usually sits at the right end of the glyph window toolbar and shows the current coordinates and other parameters of the cursor:

To the right of the meter button you will find a *zoom selection menu*:
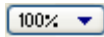
If you click on it you will get the zoom menu that has same options that you may find in the zoom toolbar. This menu is useful if zoom toolbar is not visible.

We will return to a more detailed description of the glyph window properties in the "*Glyph Window* (on page 139)" chapter.

Finally, let's quickly preview the last window in TypeTool: the Metrics window.

Metrics Window

The Metrics window is used to adjust glyph metrics — glyph sidebearings and kerning.

To open the Metrics window select some glyphs in the Font window and click on the **New Metrics Window** command in the **Window** menu.

You will see a new window:
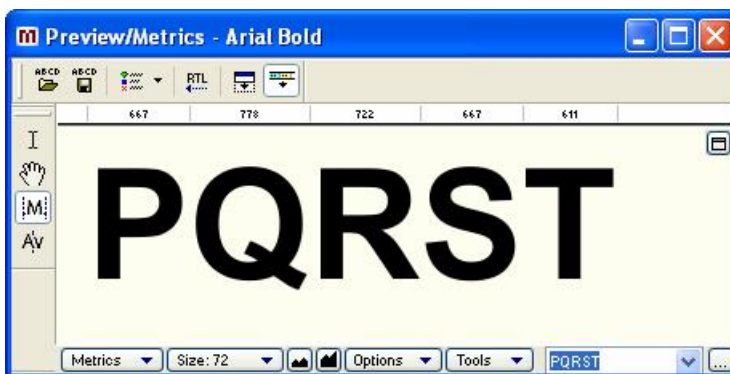
Glyphs that are currently selected in the Font window or the glyph that is in the active glyph window will appear in the Metrics window.

The Metrics window has a main editing field, a command area and two local toolbars.

To choose a string of characters to preview or modify use the string selection control:
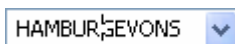
To the right of the button there is an options ⬜ button. Click on it to get access to the list of strings where you can customize it.

Metrics Window Toolbars

The Metrics window contains two local toolbars and a command area.

A Metrics window toolbar with controls for importing and exporting metrics files, automating metrics or kerning generation and other commands:

By default the toolbar is docked to the top of the window, but you can drag it to the bottom or leave it floating around.

A Metrics Tools toolbar with four buttons that allow you to select one of the metrics tools:

By default this toolbar is vertically aligned and docked to the left side of the window. You can drag it anywhere or dock to any side.

A local command area that is used to select a mode for the Metrics window and a string for metrics or kerning editing:

The local command area of the Metrics window may be located in the bottom (default) or top area of the window. When the local command area is in the top location, it includes controls to modify metrics or kerning:
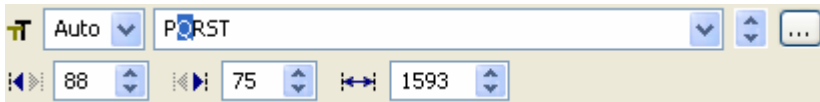
The content of this properties area depends on the current mode of the Metrics window.

Metrics Modes

The metrics window works in four modes: *text*, *preview, metrics* and *kerning*.

In Text mode you can enter and edit text in the main editing area of the Metrics window. It works very similar to any standard text editor:

Preview mode is used to preview text with kerning applied and check it at different sizes. Also the position and width of the underline and middle-stroke line can be adjusted in this mode:
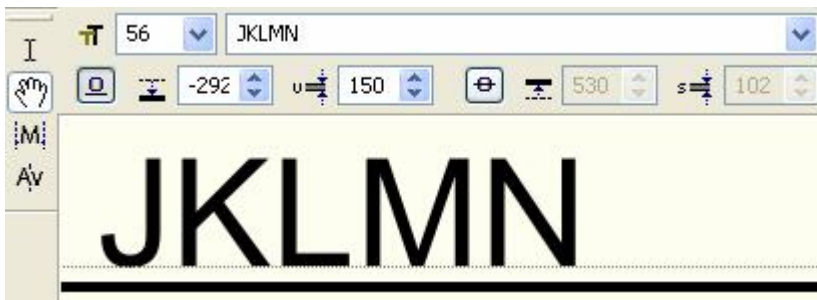
In Metrics mode you can change the glyph sidebearings using either visual or digital controls:
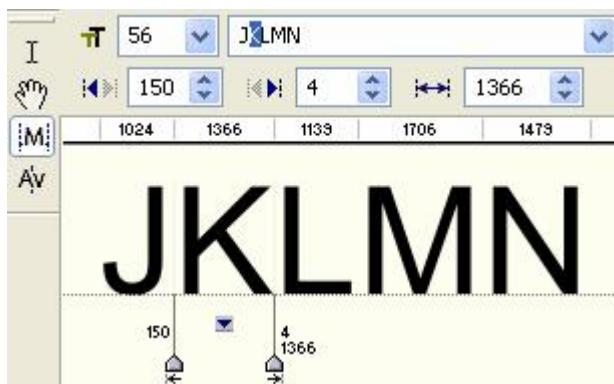
In Metrics mode the string of glyphs is previewed without kerning.

In Kerning mode you can edit pair kerning:

Metrics Panel

The Metrics Panel is a horizontally oriented table that may appear above or below the editing area:

N ▶	H	A	M	B	U	R	G	E	V
↔	853	743	981	660	746	697	818	690	706
⊢	35	-1	8	31	8	52	48	46	28
⊣	37	0	31	64	7	-13	36	29	-19
Ke								-25	

You may control the appearance of the Metrics Panel using the **Panel** command in the **Options** local menu (when the local command area is at the bottom) or with the **Panel** button on the Metrics window toolbar: ▾.

Click on this button in the top-right area of the panel to move it top or bottom:

When the Metrics Panel is visible, the properties area of the command area (if it is at the top) disappears.

Panels

Some TypeTool operations are accessible through Panels — small windows that are located in front of the main Font, Glyph and Metrics windows.

Use the **Window** menu to open panels:
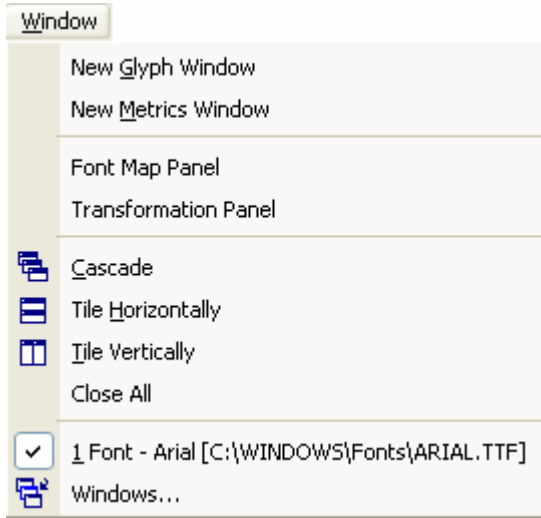
Below is the list of all the panels available in TypeTool. They are described in full detail in the sections that are related to their functions, so this is only a short reference:

Transformation	Panel for digital outline transformations
Font Map	A image representation of big Unicode fonts

All panels are described in full detail in the following chapters when we discuss the features that they serve.

Most of the panels can stick to the edge of the TypeTool window and to each other, so you can easily arrange them to create the most comfortable environment.

Every time you exit TypeTool the positions of all toolbars and panels are stored in the **Workspaces** folder, so when you run TypeTool the next time, the environment will be restored.

To reset your current workspace to the default (factory) state, hold **Ctrl** while starting the application. A dialog box will appear:
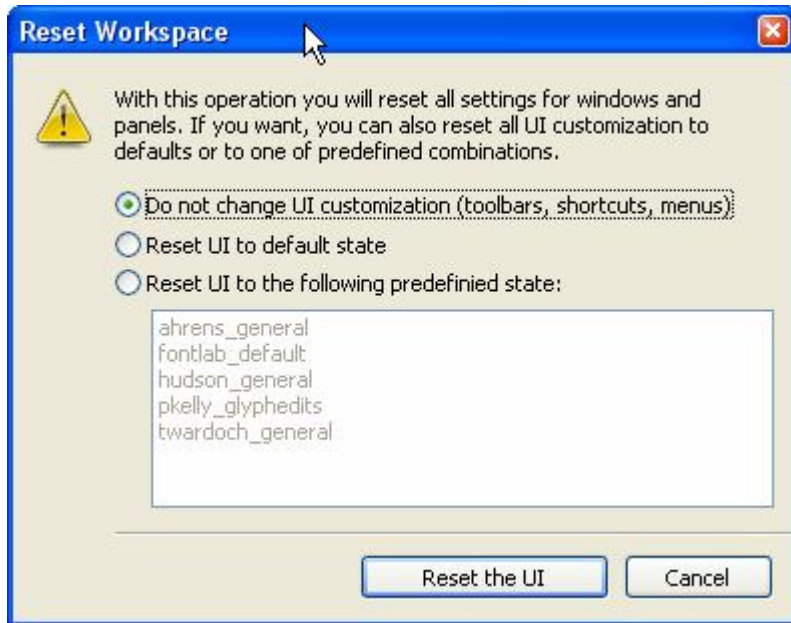
and you will be able to choose to start TypeTool with the current or the default UI.

TypeTool Options

Most of the features, behavior, import and export algorithms of TypeTool are customizable in the Options dialog box. In TypeTool 3.0 the Options dialog box has been significantly expanded. There are more options, so there are more choices. We encourage you to experiment with the settings and adapt them to your preferences. However, note that the authors have carefully chosen the factory settings so if you do not feel like poking around the Options, in most cases you will be fine with the defaults.

To open the Options dialog box, select the **Options** command in the **Tools** menu:
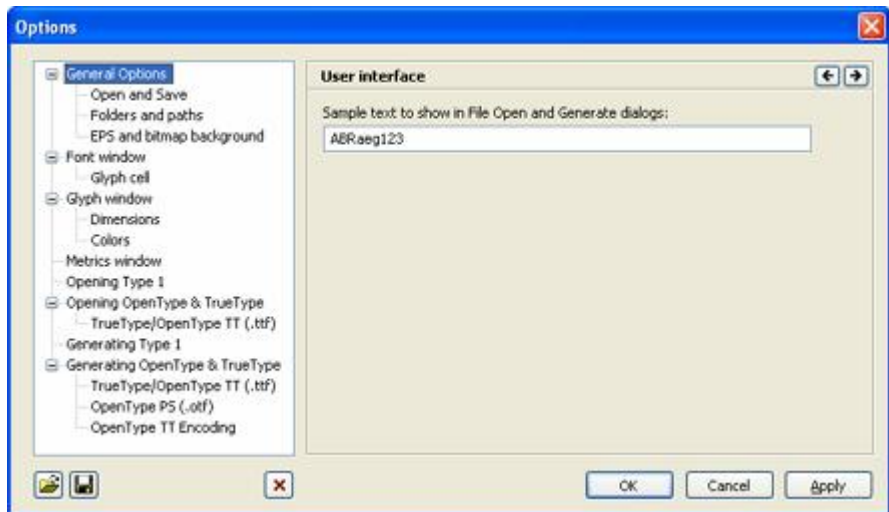
The dialog structure is quite simple. There is a list of pages combined in categories on the left, the contents of the currently selected page on the right and some buttons on the bottom. You will notice that the structure of this dialog bears resemblance to the structure of the Font Info dialog.

To select a page use the list on the left:
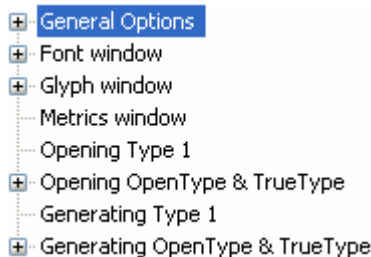
Expand one of the categories to see all the pages:
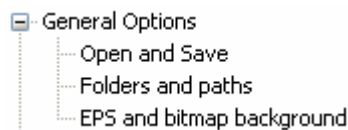
Select a page and you will see its contents appear at the right of the list:
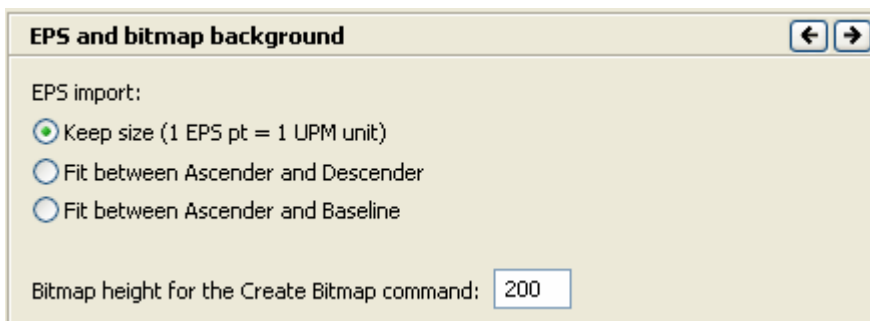
You can browse pages continuously by clicking on ⬅➡ buttons.

Other buttons and their meaning are described in the table:

	Import options	Allows you to select a profile file that holds a particular configuration of all options and loads that profile. You can create different profiles for different occasions and load them when needed — for example, separately for each format or foundry that you work with
	Export options	Exports current options to a profile file. In a workgroup environment, you can export a profile file and give it to your colleague who then can load it and generate fonts in the same environment. When sending technical problem reports to Fontlab Ltd., please always export your options into a profile file and attach that file with your report
	Reset options	Resets all options to the factory defaults
	Apply	Applies the changes without closing the dialog box. Many interface changes become visible immediately in the corresponding windows
	Cancel	Closes the dialog box without applying changes
	OK	Applies the changes and closes the dialog box.

General Options

User interface

Sample text to show in File Open and Generate dialogs:

ABRaeg123
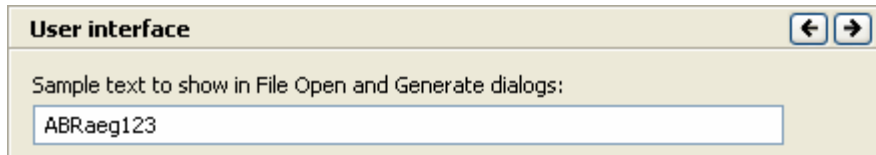
The Sample text string is used to preview the font in the Open and Generate dialog boxes. You can enter any string of text here.

Open and Save

If you want to protect yourself from system or program crashes you can use the Autosave function that will periodically save the current font.

☐ Autosave fonts every 10 minutes
☑ Create backup files
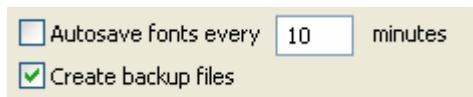
Use the check box to activate Autosave and enter the time interval (in minutes) at which you want to save the font. The font will be saved into the Autosave folder (within the Application user data folder) and will be named using the following structure:

flsX.save.vfb, where fls are the first 3 letters of Font Name and the X is some unique value.

If Autosave was active and you have a system or program crash, you can open your last saved font from the Autosave directory.

When you manually save your font and the **Create backup files** option is enabled, TypeTool will save the previous version of your font in the same folder as the currently saved .vfb file but will use the .bak file extension instead. If you would like to go back and open the previous (backup) version of your .vfb file, use **File > Open**, navigate to the folder in that you saved your file and type in ***.bak** in the File name field, then press **ENTER**. You will then see the backup file and will be able to open it.

Folders and Paths

Please refer to the "***Folders and Paths*** (on page 25)" section of the "***TypeTool User Interface*** (on page 19)" chapter for information about these settings.

EPS and Bitmap Background
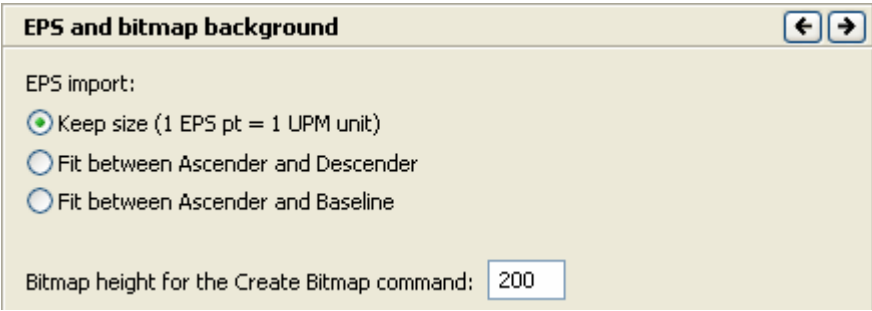
Keep size	When enabled, pasted and imported EPS/AI outlines will not be scaled, with the assumption that 1 pt in the EPS/AI drawing corresponds to 1 font unit in TypeTool
Fit between Ascender and Descender	When enabled, pasted and imported EPS/AI outlines will be automatically scaled to fit between the Ascender and Descender lines of the font
Fit between Ascender and Baseline	When enabled, pasted and imported EPS/AI outlines will be automatically scaled to fit between the Ascender and the Baseline of the font
Bitmap height for the Create Bitmap command	Allows you to set the bitmap size in pixels that will be created when the user chooses **Tools > Background > Create**. Higher values will give you a more high-fidelity bitmap rendition of your glyph but will result in larger .vfb files.

Please refer to the "***Importing and Exporting Glyphs***" section of the "***Glyph Window***" (on page 139) chapter for more information about using the first option.

Font Window

The options on the **Font Window** page control display aspects of the Font Window and define how certain commends work.

Double-click on opens Glyph Window	If enabled, double-click on a glyph cell in the Font Window opens a Glyph Window. If disabled, double-click on does not yield any action
Double-click on opens a new window	If this option is enabled, each double-click on a cell on a glyph cell in the Font Window opens a new Glyph Window. If disabled and there is already a Glyph Window open, the glyph that the user double-click on will be displayed in the existing Glyph Window. This option works only if the previous option is enabled
Enable drag-and-drop	When enabled, drag-and-drop operations work in the Font Window. Drag-and-drop in the Index mode physically rearranges glyphs in your font. Drag-and-drop in other modes of the Font Window is used to assign new code positions to existing glyphs. Drag-and-drop between fonts can be used to append (if in Index mode) or copy (other modes) glyphs between fonts
Sorting glyphs that are out of encoding	Controls the way glyphs are displayed in the Font Window that are shown outside of the "yellow area", i.e. the glyphs that do not belong to the currently selected encoding or codepage.

Glyph Cell

These options control the appearance details of glyph cells in the Font Window.

Each cell should have dimensions of...	Controls the default size of the glyph cells in all Font Windows. Note that when the local control area of the Font Window is placed at the bottom, you can use the **Increase/Decrease cell size** buttons to temporarily change the size of the glyph cells in the currently active font
Give each cell a caption	Shows/hides the caption of the glyph cell (the small rectangular bar shown at the top each glyph cell) and allows you to choose a font that should be used there.

Glyph Window

This section controls the behavior of the Glyph Window.
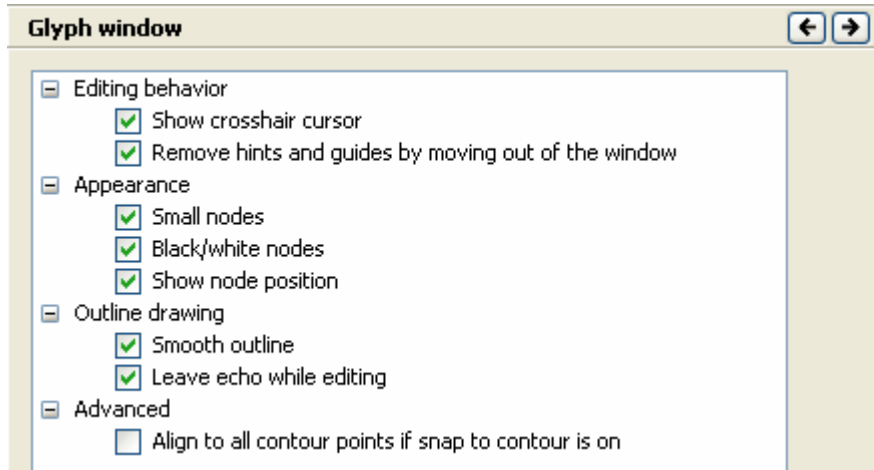
Editing behavior:

Show crosshair cursor	If enabled, a crosshair cursor is shown whenever the user moves any nodes
Remove hints and guides by moving out of the window	When enabled, the user can remove hints and guidelines by moving them out of the window.

Appearance options:

Small nodes	Nodes may be small or large:

Black/white nodes	When disabled, node symbols are displayed using color as in FontLab 3.x. When enabled, node symbols are displayed using color as in Fontographer
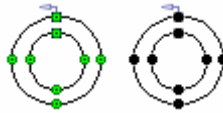
Show node position	One node may be selected as the current node. It will be highlighted and its position will appear on screen:

(446, 211)

To deselect the node, click anywhere in the empty space of the editing field or click on the **Esc** key.

Outline drawing options:

Smooth outline	Allows one to select between non-anti-aliased and anti-aliased rendering of the outline:
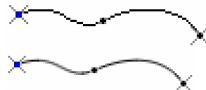
Leave echo while editing	When editing contours the original contours shape/position is shown gray:

Advanced options:

Align to all contour points if snap to contour is on	When disabled, **View > Snap to Layers > Outline** makes nodes snap only to other nodes. When enabled, the nodes snap to all locations on a contour, not just nodes.

Dimensions

These settings control visual dimensions in the Glyph Window:

Visual ascender and descender	When you select 100% as the zoom value in the Glyph Window, TypeTool needs to choose a scaling factor to fit the font unit space in the Glyph Window. This is done by always fitting the Visual ascender is to the top of the Glyph Window and fitting the Visual descender to the bottom of the window. If you think that the 100% zoom level shows you a too small portion of your glyph (because for example your font has extremely long ascenders and descenders), you can increase these values. This is only a visual setting and does not modify and metric information in the font
Grid step	This setting controls the grid step in font units. You can show/hide the grid and enable snapping to grid from the **View** menu
Snap-to distance	If any of the editing layers has the "snap-to" property turned on, moving a node will cause it to snap to the objects on that particular layer if the distance between the node and the object is not larger than the distance (in pixels) specified here. **Tip: Enabling "snap to grid" and increasing the snap-to distance may be helpful when designing pixel fonts.**

Colors
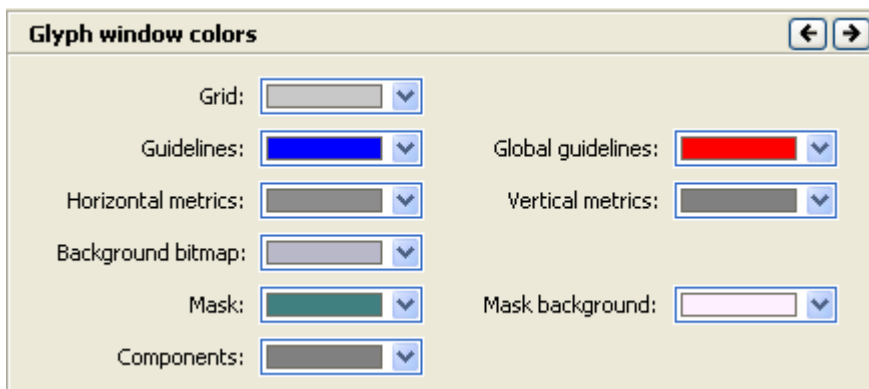
In TypeTool, the color of many elements of the Glyph Window can be customized. Click on the color box to change the element's color.

Metrics Window

These settings control the behavior of the Metrics window:

Automatic line feed	When disabled, all glyphs in the Metrics Window are displayed as a long line of text unless a line break is inserted explicitly by the user (\n). When enabled, the Metrics Window has an automatic line feed so glyphs are moved to the next line to fit within the current size of the Metrics Window
Highlight all kerning pairs	When enabled, all glyph combinations that have kerning pairs defined will be highlighted in the Metrics Window using a short horizontal line below the glyphs. When disabled, the line will not be shown
Background	Changes the background color of the Metrics Window
Foreground	Changes the color of the glyphs displayed in the Metrics Window
Font to use in the preview combo box	Allows you to select the font for use in the preview combo box.

Opening Type 1

These settings control what happens when you open a Type 1 font in TypeTool.

Opening PostScript Type 1	← →
☐ Decompose all composite glyphs ☑ Generate Unicode indexes for all glyphs	
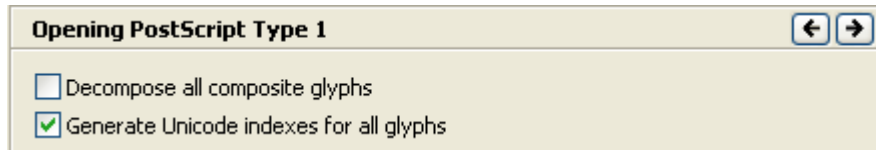
If the option **Decompose all composite glyphs** is on, TypeTool will decompose all composite glyphs in the imported font. Composite glyphs have no unique outline themselves, but "borrow" outlines from other font glyphs. Good examples of composite glyphs are accented glyphs, like 'A', 'a' or 'n'. In each of these the composite glyph is composed of a basic glyph outline and an accent glyph outline from elsewhere in the font. TypeTool has all the necessary tools and operations to work with composite glyphs, so it is usually not necessary to decompose them on import. But if you want to significantly modify the glyphs and do not want to worry about composites you can use this option. Of course you can always decompose or recompose the glyphs later using TypeTool commands.

The option **Generate Unicode indexes for all glyphs** should be usually on. We strongly recommend keeping it that way if you plan to convert your Type 1 font to TrueType or OpenType format. TrueType and OpenType formats uses Unicode codepoints to access glyphs, so having the indexes set properly is paramount. However, if you do not plan to make a TrueType font you may switch this option off. As in the case of the first option, you can always make Unicode codepoints later.

How TypeTool Makes Unicode codepoints

TypeTool uses a file STANDARD.NAM which is a mapping file that contains a list of PostScript names and corresponding Unicode codepoints.

When you import a Type 1 font and the option **Generate Unicode indexes for all glyphs** is on TypeTool takes the name of every imported glyph and looks for it in the names database. If it locates the name there it takes the associated Unicode codepoint and adds it to the glyph's list of indexes.

Note 1: The Names' database has more than 4000 records and includes almost all known names for all European, Cyrillic, Arabic and Hebrew languages and for most Symbol and Dingbats fonts.

Note 2: The names' database is a text file that can be edited. You can add new records to this file at any time. Be very careful when you edit this file because incorrect records may make exported fonts unusable in some environments.

Note 3: It is possible to link more than one Unicode codepoint to a name. If TypeTool finds several indexes linked to the name, it will assign all the indexes to the glyph. (Refer to the Encoding Modes section for a description of the multi-Unicode codepointing method.) For glyph names preceded with "!" in the mapping file, TypeTool will generate Unicode codepoints based on these glyph names.

Opening OpenType & TrueType

These settings control what happens when you open a TrueType ∕ OpenType TT (.ttf) or an OpenType PS (.otf) font in TypeTool.

Opening OpenType PS, OpenType TT and TrueType	← →
☑ Scale the font to 1000 UPM	
☐ Decompose composites	
☐ Store custom TrueType/OpenType tables	

Scale the font to 1000 UPM	Typically TrueType fonts have UPM (Units Per eM — the size of the grid on which all glyph coordinates are defined) equal to 2048. Type 1 fonts have UPM equal to 1000. You can change the UPM value at any time using the TypeTool commands, but if you turn this option on, UPM will be converted during the font import
Decompose composites	When enabled, all composite glyphs will be automatically decomposed. Refer to the previous section for more information about automatic decomposition. **Note: When TypeTool opens TrueType / OpenType TT fonts with rotated or slanted components, it will always decompose them**
Store custom TrueType/ OpenType tables	Some TrueType fonts have additional tables that are not a part of the TrueType or OpenType specification. If you want to read these tables and have them written in an unchanged form into the generated font, enable this option.

TrueType/OpenType TT

These settings only apply when you open a TrueType / OpenType TT (.ttf) font but not an OpenType PS (.otf) font.

Convert TrueType curves into PostScript curves	In TypeTool, you can work with PostScript Bézier curves or TrueType quadratic curves. If you open a TrueType / OpenType TT font and plan to generate the font in the same format, disable this option to keep the original outlines to avoid conversion errors. But if you plan to generate your font as a Type 1 or OpenType PS font, you can enable this option to convert the outlines on import. In any case, you can always convert the outlines in either direction at any time during editing
Store TrueType native hinting	Leave this option on if you want to store the original TrueType instructions and outlines. TypeTool will keep the stored TrueType data until you change the glyph's outline or hints. If you open a TrueType font to rearrange glyphs or to add some new glyphs we highly recommend storing the original TrueType hinting data
Autohint font	To prepare an imported TrueType font for Type 1 editing and export, you may ask TypeTool to automatically make Type 1 hints for all the glyphs. TypeTool will use the current Type 1 hinting settings and will make hints for TrueType or Type 1 outlines depending on the conversion setting (Convert TrueType curves into PostScript curves).

Generating Type 1

These settings control some technical parameters of fonts that you generate in the Type 1 format.

Make PFM file	Switch this option on to create a PFM (Printer Font Metrics) file when exporting a Type 1 font. PFM files are used in Windows to install Type 1 fonts. They contain metrics, kerning and, partially, font header information. On Windows, you cannot install a Type 1 font without a PFM file. If you have Adobe Type Manager 4.1, it can automatically generate a PFM file based on an AFM and INF file but in general, we recommend leaving this option on at all times
Make AFM and INF files	Switch this option on to make AFM (Adobe Font Metrics) and INF (font INFormation) files when exporting a Type 1 font. These files are text files and contain descriptions of the font metrics, kerning and header (font names, weight, width, encoding and other information). It is possible to install a Type 1 font with Adobe Type Manager if you do not have the PFM file but do have the AFM and INF files, because ATM will automatically build the PFM file using data from the AFM and INF files.
	The AFM file is necessary to install a Type 1 font (in ASCII form, with ".pfa" extension) in most Unix-based operating systems.
	To install an exported Type 1 font in Windows you must have the PFM or at least AFM+INF files. We recommend making all these files when you finally produce a font so that your font will be compatible with various environments.

The dialog has a list of possible encoding options when a Type 1 font is generated:

Encoding options:

Export Unicode codepage if codepage mode is active

Always write custom encoding
Always write Standard Encoding
Export Unicode codepage if codepage mode is active
Select encoding automatically
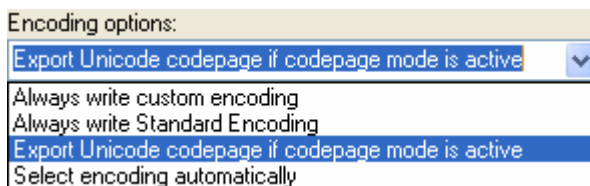
When you generate a Type 1 font, the most important encoding choice is to choose between one of two encoding forms:

• Standard Encoding
• custom encoding

Standard Encoding is a special Type 1 encoding created by Adobe Systems. Instead of enumerating all the code positions in the font, the Standard Encoding font leaves the actual encoding to the system font driver, and on the other hand, the system font driver knows what characters can be expected in the font. StandardEncoding is the recommended choice if your font is a typical Western Roman font. If you generate a Mac Type 1 font (on TypeTool for Mac) with Standard Encoding, Mac OS will "know" that the font is a standard Western Roman font and will automatically match the font's encoding to the system Mac Roman codepage. Similarly, when you generate a Windows Type 1 font with Standard Encoding, Windows will know that the font is a standard Western Roman font and will automatically match the font's encoding to the system Windows 1252 Western (ANSI) codepage. Also, if the user of such fonts create documents in some applications (e.g. QuarkXPress for Mac and Windows), the applications will automatically re-encode the documents when moving between platforms.

Custom encoding is any Type 1 encoding which is explicitly specified in the font. If the primary character set of your Type 1 is not Western Roman but Central European, Cyrillic, Greek, or Arabic, you need to select the appropriate encoding in the Names mode in Font Window, and generate the font with custom encoding.

73

How Windows ATM Interprets a StandardEncoding

When a Type 1 font has StandardEncoding ATM assumes that this font includes all the glyphs from the first 128-glyph range (digits, alphabet and basic punctuation) and the European glyphs (128-255 range). The first 128 glyphs are called the "*top zone*". The 128-255 range is called the "*bottom zone*". The Adobe StandardEncoding includes very few glyphs from the bottom zone compared to the number of glyphs in the WinANSI (actual Windows encoding) encoding. When a Type 1 font in StandardEncoding is installed with ATM, ATM uses a special encoding instead of the "real" StandardEncoding as it is documented in the Type 1 format specification. This special Windows encoding is called the Default Encoding in TypeTool. So if you create a StandardEncoding font and want to see how it will work in Windows, select the Default Encoding in TypeTool.

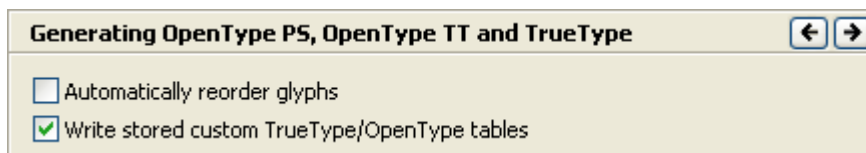## Here is an explanation of the possible encoding export options:

Select encoding automatically	This is the recommended setting. Generates the font with Standard Encoding if the Font Window is in Names mode and the encoding selector shows one of the following: "Adobe Standard Encoding", "Default Encoding", "MS Windows 1252 Western (ANSI)" or "Mac OS Roman". Otherwise, a custom encoding will be generated
Always write custom encoding	TypeTool will always generate a custom encoding — even for Western Roman fonts — with the encoding currently selected in the Font window Names mode. Note: Western Roman Type 1 fonts generated with custom encoding may not work as expected
Always write Standard Encoding	Always generates the font with the Standard Encoding regardless of what is selected in the Font window Names mode
Export Unicode codepage if codepage mode is active	Exports a custom encoding based on the currently selected codepage if the Font window is in Codepages mode.

We recommend setting the **Select encoding automatically** option as the default, because it covers most exporting situations very well.

Export only encoded glyphs	When enabled, all glyphs that are outside of the encoded "yellow area" will not be included in the generated font.
	Tip: This can be used to quickly generate a series of Type 1 fonts from a large multilingual .vfb file that includes an extensive character set.

Generating OpenType & TrueType

These settings control some technical parameters of fonts that you generate in the TrueType / OpenType TT (.ttf) or OpenType PS (.otf) format:

Generating OpenType PS, OpenType TT and TrueType	← →
☐ Automatically reorder glyphs	
☑ Write stored custom TrueType/OpenType tables	

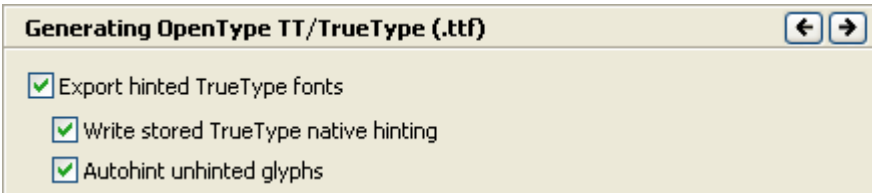|	
Automatically reorder glyphs	If this option is enabled, TypeTool will try to reorder glyphs to match the Mac cmap encoding table. Technically, this is a requirement of the Apple TrueType specification but it is not required on Mac OS X or Windows.
Write stored custom TrueType/OpenType tables	When enabled, stored custom TrueType/OpenType tables will be written into the generated font.

TrueType/OpenType TT (.ttf)

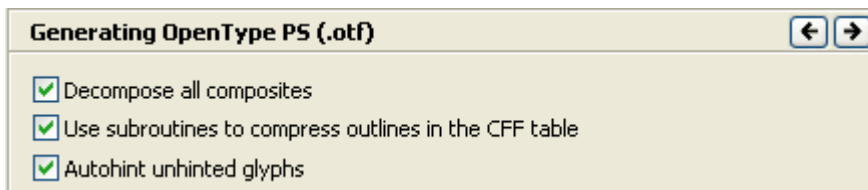These settings only apply to fonts that you generate in the TrueType / OpenType TT (.ttf) format:

Export hinted TrueType fonts	TypeTool will export TrueType instructions of any type (original or automatically generated) only if this option is on. To create a completely unhinted TrueType font switch off this option (may be useful for pixel fonts)
Write stored TrueType native hinting	If this option is on and the original TrueType instructions were stored when the font was opened, then TypeTool will try to restore the original instructions where possible. If you want to discard all the original TrueType instructions, switch this option off
Autohint unhinted glyphs	If this option is on, TypeTool will try to automatically generate TrueType instructions for all unhinted glyphs.

How TypeTool Autohints TrueType Fonts on Export

When autohinting is allowed and TypeTool finds a glyph that has no TrueType hints of any kind it begins to make TrueType hints automatically. If this glyph has Type 1 hinting information, then TypeTool converts this information to visual TrueType instructions and converts the instructions to the TrueType hinting code. If Type 1 hints are not present then TypeTool automatically generates Type 1 hints as the first step, then converts the Type 1 hints into TrueType visual instructions and converts the visual instructions into TrueType native instructions.
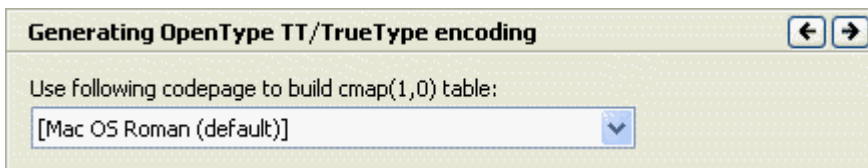
OpenType PS (.otf)

These settings only apply to fonts that you generate in the OpenType PS (.otf) format.

Generating OpenType PS (.otf)

☑ Decompose all composites
☑ Use subroutines to compress outlines in the CFF table
☑ Autohint unhinted glyphs

Decompose all composites	When enabled, all composite glyphs in the font will be decomposed. Recommended for maximum compatibility.
	When disabled, the composite glyphs will be exported as such
Use subroutines to compress outlines in the CFF table	Allow to automatically generate outline subroutines if font is generated as CFF-flavored. Outline subroutines store repetitive parts of outlines and allow to reuse with references from outline definition code
Autohint unhinted glyphs	When enabled, all glyphs that contain no hints will be autohinted.

OpenType TT Encoding

These settings control some advanced re-encoding features for fonts that you generate in the TrueType / OpenType TT (.ttf) format. These settings do not apply to OpenType PS fonts.

Generating OpenType TT/TrueType encoding

Use following codepage to build cmap(1,0) table:

[Mac OS Roman (default)]

Use following codepage to build cmap table	This is a "hack" used in older operating systems (Windows 95/98) to improve handling of single-codepage non-Latin fonts. It is not required and not recommended, but if you find that you need to use this hack to get your font to work in an old OS, switch on this option and re-export the font

Editing Fonts

In this chapter we will discuss the editing of fonts. A font is a collection of glyphs with similar design and some encoding and header information. The information includes font identification names, copyright data, character encoding information and other data that is necessary for font usage. Generating fonts is not discussed in this chapter. Refer to the "***Generating Fonts*** (on page 349)" chapter for this information.

Opening Fonts

With TypeTool you can create new fonts or open existing fonts for modification. When you open an existing font, however, please be sure that modifying it does not violate copyright laws: some fonts are copyrighted as software so it is not legal to change them in any font editor. Carefully read the license agreement that comes with every font.

You can find two (one serif and one sans-serif) royalty-free, non-copyrighted fonts that you can use as a basis for your own fonts or glyphs in the Sample folder or on our website (***Fontlab Home Page*** (http://www.fontlab.com/)).
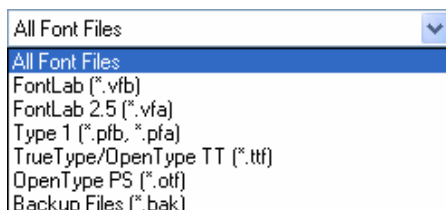
To open a font for editing, select the **File > Open** command, or click on the button on the toolbar.

You will see the Open File dialog box in which you can select a font file to open. In this dialog box, you will see all the fonts that can be opened: TrueType/OpenType TT (.ttf), Windows TrueType Font Collection (.ttc), Windows Type 1 (.pfb), Unix/ASCII Type 1 (.pfa), OpenType PS (.otf), FontLab 2.5 font files (.vfa), TypeTool and FontLab 3.x/4.x/Studio 5 font files (.vfb).
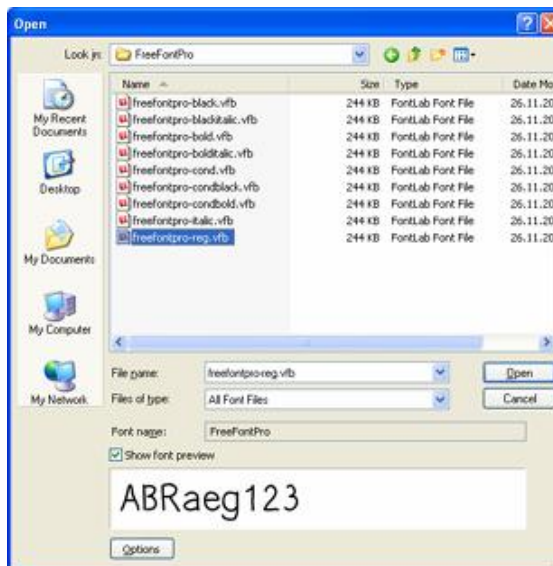
If you want to quickly open a font from the list of the most recently opened fonts, select that font in the **File name** combo box.

If you want to list only fonts in a particular format, select that format in the **Files of type** combo box:

| All Font Files	∨
All Font Files	
FontLab (*.vfb)	
FontLab 2.5 (*.vfa)	
Type 1 (*.pfb, *.pfa)	
TrueType/OpenType TT (*.ttf)	
OpenType PS (*.otf)	
Backup Files (*.bak)	

When you select a font file in the files area, you will see the font name and preview below. Switch off the **Show Font Preview** option to not see the font preview.

You can open many fonts with a single operation: select all of them in the list with a selection rectangle or using **CTRL**-click.

You can set the opening options here by clicking on the **Options** button. Please refer to the "*TypeTool Options* (on page 56)" section for a detailed discussion of the opening options.

You cannot use **File > Open** to open fonts located in the system font folder (typically C:Windows\Fonts). To open any of these fonts, use **File > Open Installed**. This will show a dialog box that displays all fonts installed on your system — choose one and click on **OK** to open the font.
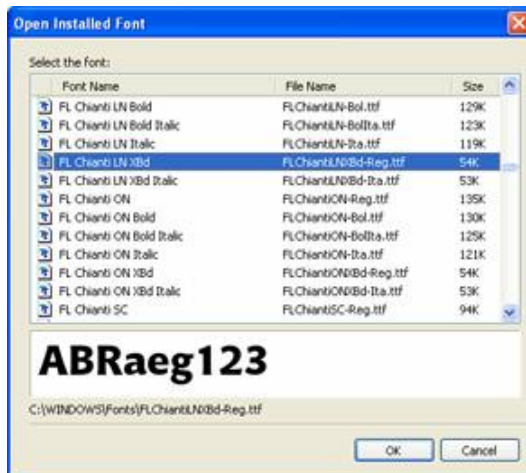
Most Recently Used Fonts

All fonts that you recently opened in TypeTool are added to the list of the most recently used font. This list is used in the File Open dialog and also at the bottom of the **File** menu:
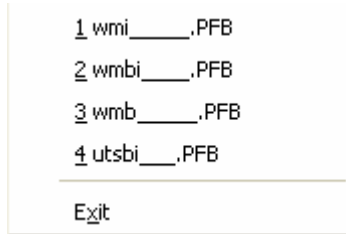
Next time you wish to open any of them, select the font file in the **File** menu and TypeTool will open them.

Opening Fonts with Drag-drop

An easy way to open fonts in TypeTool is to drag-drop font files from Windows Explorer onto the TypeTool application window:
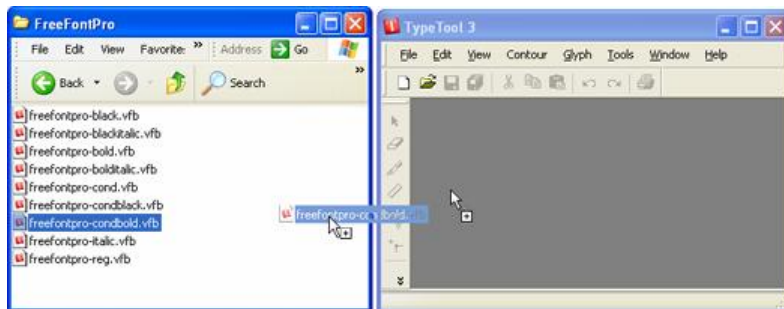
Font Formats

The .vfb file format used in TypeTool 3.0 is fully backwards-compatible, so TypeTool 3.0 can open any .vfb file created in TypeTool 1.x and 2.x. The format is also cross-platform-compatible so .vfb files saved from the Windows version can be opened in the Mac version and vice versa. In addition, the format is largely upward-compatible. This means that a .vfb file saved from TypeTool 3.0 can be opened in FontLab 3.x — 5.x, as well as other Fontlab Ltd. products such as **_TransType_** (http://www.fontlab.com/transtype/) or **_AsiaFont Studio_** (http://www.fontlab.com/asiafontstudio/). Only those elements of the format that were supported by the old version will be retained and some information may change slightly. However, the most important elements of the font such as key Font Info entries, glyph outlines and kerning pairs will be retained.

For example, .vfb files saved from TypeTool 3.0 for Windows can be opened in FontLab Studio 5.0 for Macintosh and vice versa, with as much as possible information retained.

TypeTool 3.0 also opens .vfa files saved from FontLab 2.5 (but not 2.0). If you have fonts saved in a proprietary format of another application and would like to open these fonts in TypeTool, the best way is usually to create a Windows-compatible Type 1 font from your other application and open the Type 1 font in TypeTool. If you wish to move your .fog files created in **_Fontographer_** (http://www.fontlab.com/fontographer/)3.5 or 4.1 to TypeTool, you can use our **_FogLamp_** (http://www.fontlab.com/foglamp/) product that converts Fontographer .fog files into TypeTool-compatible .vfb files, retaining not only outline information but also mask layers, guidelines, background bitmaps etc.

Multiple Master Fonts

Multiple Master fonts contain several font styles, called *master fonts*, in one font file. The Multiple Master font format is an extension of the Type 1 font format. A program that uses a Multiple Master font can select not only one of the master fonts, but also any intermediate style created by interpolation of the master fonts. So it can use not only Bold, Normal, Narrow or Wide styles, but any style in between, such as 30% Weight and 47% Width.

With TypeTool you **cannot** open and edit Multiple Master fonts. Instead you have the possibility to create a regular single master font from an instance of a Multiple Master font. When you select a Multiple Master font in the File Open dialog box the **Choose Instance** dialog box appears:
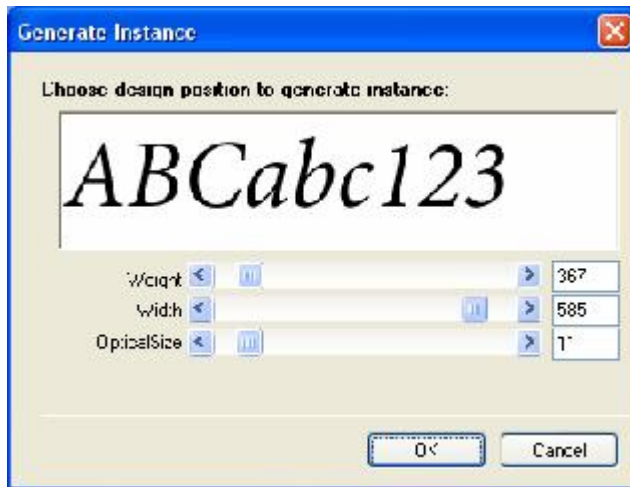
Use the controls to select the intermediate design. Click on **OK** and TypeTool will generate a **single master** Type 1 font and open it in a new Font window.

If you need full support for Multiple Master fonts, we recommend you try *FontLab Studio* (http://www.fontlab.com/studio/).

Importing Font Collection

When you select a Windows TTC (TrueType Collection) font for open, you are presented with a special TTC Font Collection Import dialog box:

The list of fonts in a collection has check boxes to let you select which fonts should be imported. You can check those of them that you want to be opened. Click on the **Select All** button to switch on every check box. To switch them all off click on the **Unselect** button.

When you are ready click on the **OK** button to start importing TTC font.

Creating a New Font

If you want to create a new font from scratch, you select the **New** command from the **File** menu. TypeTool will create an empty font that will not have any glyphs and will open an empty Font Window.

When you have created a new font, it is a good idea to first go to **File > Font Info > Names and Copyright**, fill in the Family Name and the Style Name (even if they are temporary). Then press **Build Names** (you can fill in the remaining Font Info entries later). Then click on **OK**, choose **File > Save As** and save the new font in the .vfb format under a new filename in a folder of your choice.

Now you can start to create your glyphs (in the Font Window), design them (in the Glyph Window), letterspace them (in the Metrics Window). Also, you should fill in the important Font Info fields (see the "***Font Header*** (on page 313)" chapter for details). When this is done, you can generate your font in the font format of your choice, e.g. OpenType PS (.otf), install it on your system and test it.

The Font Window

The Font Window is used to display an entire font. It opens automatically when you open an existing font for editing or choose to create a new font.

In TypeTool you can open many fonts at once and every font will have its own Font Window. The Font Window is a representation of the font, so when you close this window the font will also close.

You can do a lot of things using the Font Window — from browsing a font for a desired glyph to rearranging and remapping the font to editing the Font info fields. The following sections of this chapter will tell you how to use this window.
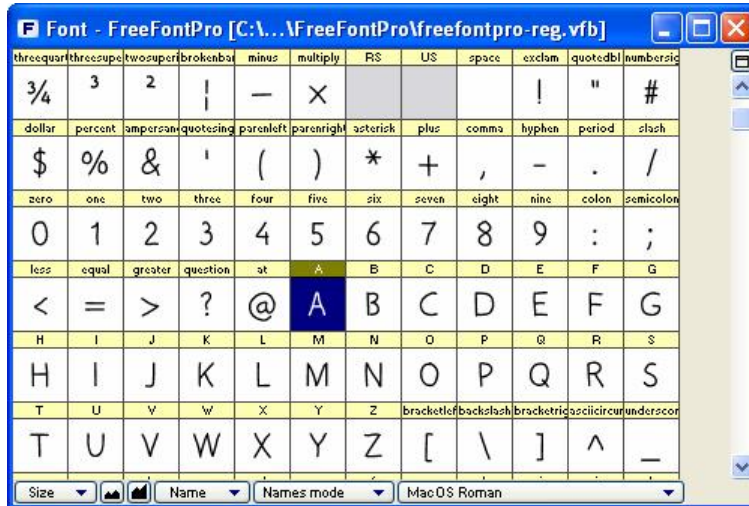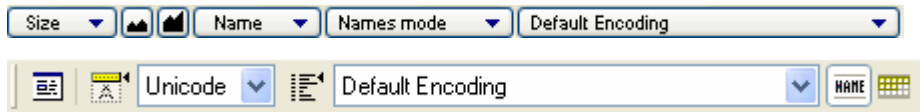

The Font Window consists of the *command bar* at the bottom, and a glyph *table,* where a single cell represents each glyph:
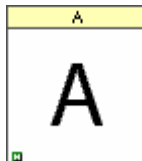
Each cell has a *caption* at the top that shows some identification information — it may be the name of the glyph, its code in various forms or some other glyph information.

➲ Tip: You can change the font size used to display the caption in **Tools > Options > Font Window > Glyph Cell**.

The Font Window command bar has two alternate forms — it can be placed either at the top or at the bottom of the Font Window.

You can switch between the top and the bottom location of the command bar by clicking on this button ▣ in the top-right corner of the Font Window.

The left combo box located on the Font Window command bar (if in top position) or the Caption context menu (in the bottom position) lets you select one of the caption modes:
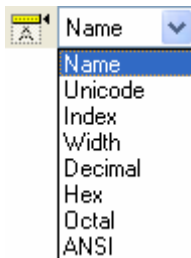

Depending on the selection, a different text string will appear in the caption:

Name	The glyphname (the so-called PostScript name of the glyph)
Unicode	The Unicode codepoint assigned to the glyph, in hexadecimal form
Index	The glyph index, i.e. the physical location of the glyph in the font
Width	The glyph's advance width
Decimal	The local character code in decimal form
Hex	The local character code in hexadecimal form
Octal	The local character code in octal form
ANSI	The ANSI character that corresponds to the local character code.

The glyph cells may have different colors. The background of the glyph cell may be grey or white, and the caption may be white or yellow.

A grey cell background means an empty glyph. This means that the glyph does not exist in the font and that the glyph cell is displaying a glyph placeholder. The placeholder usually consists of a glyph template image.

TypeTool 3.0 ships with a very extensive set of pre-installed default glyph template images. These are based on the Andale Mono WTG font (courtesy of **Monotype Imaging** (http://www.monotypeimaging.com/)) and cover the entire Unicode 3.2 character set. Note that the default glyph template images are low-resolution, monospaced and in a "sanserif" style. They should not be used as direct source of information about the typographically correct shape of glyphs — but only as an orientation.
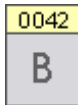
A white cell background means that a glyph exists in the font.

If the glyph cell background is white and there is no image in it, we speak of a blank glyph. A blank glyph means that the glyph exists in the font but does not contain any outlines or components. If the white cell includes a pale grey image, it means that the glyph there is a bitmap background in the glyph but no outline or components.

If the cell includes a black image, it means that the glyph exists and is non-blank, i.e. it contains an outline or a component.

A yellow caption of a glyph cell means that the glyph is part of the currently selected encoding or codepage, or, as we say, is "in the yellow zone" (see next section). Glyphs that are not part of the current encoding have a grey caption.

The small blue mark that appears in the left-top of the glyph cell means the glyph has more than one Unicode codepoint assigned:
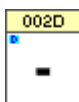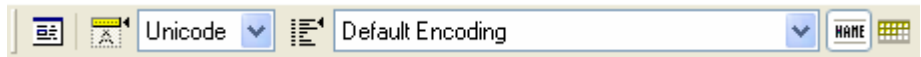

When you modify a glyph in any way, a black bar below their caption appears. The black bar indicates glyphs modified since the last save. When you save a font, all black bars disappear.

The "E" glyph has been modified.

Font Window Command Bar

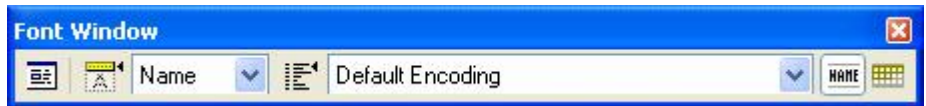On the Font Window command bar in the top position you see one button on the left and two buttons in the right area:

The left button ⊞ opens the Font Info dialog box for the current font. This is the same as choosing the **Font Info** command in the **File** menu.

Buttons on the right allow you to select one of the encoding modes:

|	
HAME	Switch to the Names mode
▦	Switch to the Codepages mode

The Font Window command bar is not fixed to the top edge of the window. You may drag it to any place on the screen or "dock" it to the bottom of the window:

You may also close it at all to get more space for glyphs on your screen.

You can switch between the top and the bottom location of the command bar by clicking on this button ▣ in the top-right corner of the Font Window.

In the bottom position of the Font Window command bar, the bar does not have the **Font Info** button.
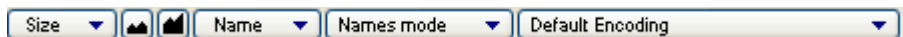
It does however contain a **Size** context menu that allows you to temporarily change the size of the glyph cells in the current Font Window. Possible sizes vary from 16x16 up to 128x128 pixels. Smaller cells occupy less space but hide details. If you select the smallest size (16x16) you will not be able to see the additional marks which are visible in cells at larger sizes.

91

You can also use the next two buttons to decrease (⬛) or increase (⬛) the glyph cell size in the current Font Window.

A sample of the different cell sizes:

16x16   24x24   32x32   48x48   64x64
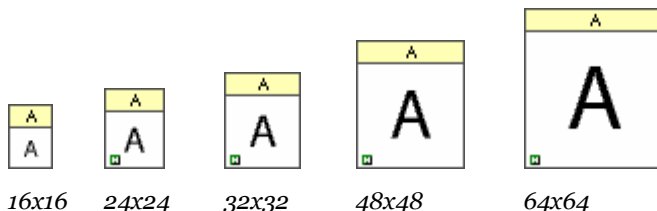
To permanently change the size of glyph cells in all Font Windows, go to **Tools > Options > Font Window > Glyph cell > Each cell should have dimensions of**, and set the size of the glyph cells.

As mentioned previously, the **Caption** context menu allows you to choose the caption text. The next combo box allows you to switch between the different modes, and is equivalent to the two buttons in the top position:

The third context menu is equivalent for the **Encoding** menu in the top position of the command bar, and is discussed later. The current glyph name and Unicode codepoint as well as the total quantity of glyphs are shown in the right part of the command bar in the bottom position.

Glyph Naming and Character Encoding

TypeTool supports different character encoding methods: the international Unicode Standard as well as legacy or specialty codepages and custom encodings.

Here's how it works:

A font is a collection of glyphs that are used to represent characters (more about that in the following section). On an average screen the Font Window can show just a few hundred glyph cells, so we need to have some method to browse the font "through" the Font Window. And on the other hand, different font formats use different methods to encode characters.

In TypeTool you can choose one of two so-called Encoding modes that allow you to select a subset of the glyph collection and show it in the top part of the Font Window for easier access. Two buttons in the Font window top command bar or the **Mode** context menu in the bottom command bar are used for the encoding modes selection in TypeTool.

In the following sections you will find more information about encoding modes, Unicode and name-based identification and the character-glyph model.

Characters, Codes and Glyphs

A font is a collection of glyphs that usually have a common design. In addition to storing each glyph, a font has some header information that stores general information about the font such as the family name, the style name, the copyright string, the ascender and descender values, and others. For more discussion about the font header information see the "***Font Header*** (on page 313)" chapter.

Simply speaking, text in digital form is a collection of *character codes* (or "*codepoints")* — integer numbers. When you enter text into a computer, the computer turns the keystrokes that you press on the keyboard into integer number and assigns a number (character code) to each character that you enter. When the computer needs to show some text on screen or print it, it accesses a font and turns the character codes into visual shapes.

A *character encoding standard* is (simply speaking) a table that defines the relation between characters and the codes that are used to represent these characters in the computer.

Character Encodings Standards

There are many other character encoding standards (sometimes called *codepages*) used in the world to help use different languages — in fact, the huge amount makes the use of the word "standard" questionable.

The main difference between the encoding standards is the size of the code. There are one-byte, double-byte and multi-byte mapping standards. With a one-byte mapping standard, each character in the text is encoded using exactly one byte (8 bits of information). This means that only 256 different characters can be encoded in a particular one-byte encoding standard.

A double-byte mapping standard uses two bytes (16 bits) for every character. So it is possible to map 65,536 characters. Multi-byte mapping standards use from one to four bytes for every character — expanding the code space to billions of characters.

The biggest problem of single-byte encoding standards (codepages) is the limited capacity. With only 256 slots (available codepoints), usually only characters from one alphabet (writing system) can be encoded. It is not possible to encode e.g. Latin and Cyrillic text in the same codepage.

Pięć flakonów wody „Экземпляръ”.
actual text

Pięć flakonów wody „Ýęçĺěďë˙đú”.
text encoded as Windows 1250 (Central European)

Pікж flakonyw wody „Экземпляръ”.
text encoded as Windows 1251 (Cyrillic)

256 character codes are not even sufficient to encode various accented (diacritic) characters from different languages that use the Roman alphabet. This is why separate codepages were created for Western European languages (English, German, French etc.), Central and Eastern European languages (Polish, Czech, Hungarian etc.), Baltic languages (Latvian, Lithuanian, Estonian etc.) and so on. In addition, different companies assigned character codes differently. For example the letter ä (adieresis) is represented using the character code 228 in the Windows Western codepage used by Microsoft and using the character code 138 in the MacOS Roman codepage used by Apple. The confusion becomes evident if you realize that the same code (138) in the Windows Western codepage is used to represent the Š (Scaron) that... does not have its own codepoint at all in MacOS Roman. On the Macintosh, it is only available in the MacOS Central European codepage, under the code 225.

The Unicode Standard

Fortunately, one predominant character encoding standard has gained popularity in the past years: the *Unicode Standard* (or short, Unicode). It assigns unique character codes (codepoints) to practically all characters used by humanity. ä has the character code 00E4 (in hexadecimal notation, which corresponds to 228 in decimal notation but for Unicode codepoints, usually the hexadecimal notation is used) and Š uses the codepoint 0160 (352 in decimal).

a	→	97	0x0061	Я	→	1103	0x044F	
á	→	225	0x00E1	א	→	1488	0x05D0	
ą	→	261	0x0105	☻	→	9787	0x263B	
α	→	945	0x03B1	練	→	32244	0x7DF4	

Unicode is a character coding system designed to support the interchange, processing, and display of the written texts of the diverse languages of the modern world. In addition it supports classical and historical texts of many written languages. Modern operating systems such as Mac OS X or Windows 2000/XP use the Unicode Standard as the default way to store text. Similarly, modern font formats such as OpenType and TrueType use Unicode to store character information.

Unicode can use up to four bytes to encode a character, it is theoretically possible to encode 4,294,967,296 characters, although the Unicode Consortium agreed that no more than 1,114,109 codepoints will ever be assigned. In the current (as of September 2005) version 4.1 of the Unicode Standard, a total of 97,786 codepoints have been assigned (less than 9% of the possible space). The vast majority of the codepoints are Asian (CJK: Chinese, Japanese, Korean) characters.

65,535 codepoints are encoded in the so-called Basic Multilingual Plane (BMP). The codepoints in the BMP are two-byte, so four hexadecimal digits are used to write the codepoint (e.g. 0160). In addition, more characters are encoded in supplementary planes. They use 5- or 6-digit codepoints, e.g. 1D56C.

Visit the Unicode Consortium official Web site for more information: ***http://www.unicode.org*** (http://www.unicode.org/)

The Character and Glyph Model

People recognize and process characters by their shapes. Thus, people normally closely associate a character and its shape. Information technology, in contrast, makes distinctions between the concepts of a character's meaning (the "character") and its shape (the "glyph"). In information technology, *characters* are abstract information elements used for data coding and interchange while *glyphs* are presentation elements used for displaying and printing the data.

Unfortunately, different literature and different standards define the border between characters and glyphs differently. For the purpose of font technology, a glyph is a single element of the glyph collection stored within a digital font file while a character is a text encoding codepoint used in text processing. Glyphs are used to visualize characters. Each font has a different glyph for the same character, for example all the glyphs: A A A̋ A A A A A are used to visually represent the same character 'A' (Unicode codepoint 0041).

In short: *characters are codes, glyphs are images*.

Even within the same font, there is no 1:1 relation between characters and glyphs. The same glyph can be used to represent two characters, the Latin letter A (Unicode codepoint 0041) and the Cyrillic letter A (Unicode codepoint 0410).

0041	0410
A	A

On the other hand, multiple glyphs can be used to represent the same character — an OpenType font can include alternate glyphs.

A	A.alt1	A.alt2	A.swash1	A.swash2	A.swash3	A.swash4	A.swash5	A.swash7
A	A	A	A	A	A	A	A	A

Characters and Glyphs in TypeTool

When a text editor displays text on screen using a font, a process of character-to-glyph mapping must occur. The application sends a request to the font rasterizer for a rendering of a character code. The font rasterizer looks up the character code in the *character mapping table* that is included in the font. The mapping table maps character codes to glyph indexes of individual glyphs. Then, the rasterizer located the glyph using its glyph index in the glyph collection of the font. Finally, the rasterizer produces the image of the glyph at a specified size and sends it back to the application.
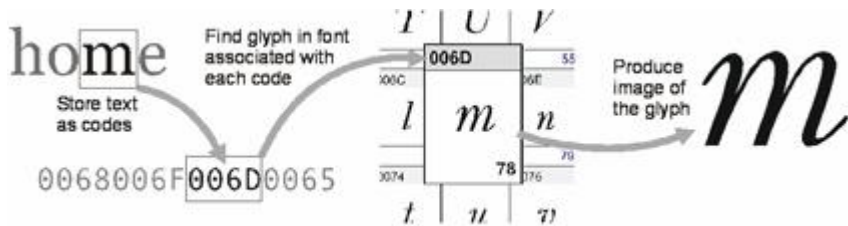
As discussed earlier, captions of glyph cells in TypeTool may display various kinds of information. The following ones represent important properties of glyphs that are all involved (one way or the other) in the character-to-glyph mapping process.

The glyph index represents the physical location of the glyph in the font's collection of glyphs. The glyph with the index 0 is physically the first glyph in the font. Some applications such as Adobe InDesign display the physical order of glyphs in a Glyph Palette so it is wise to keep control of the glyph order.

The glyphname is a short text identifier for the glyph. For example, the glyphname for the character a is a, and for the character ä is adieresis.

It is a good idea to assign meaningful glyph names to all glyphs in your font regardless of the font format. They are mandatory in Type 1, Multiple Master and OpenType PS fonts. Theoretically, they can be omitted in TrueType or OpenType TT fonts it is a good idea to use them everywhere.

In most situations, TypeTool automatically assigns glyph names to glyphs when you create a new glyph so you do not need to worry about them too much. However, many user interface elements of TypeTool use glyph names as the primary way to refer to glyphs. Therefore, you should get used to thinking about glyphs in terms of glyph names.

The Unicode codepoint is a hexadecimal number associated with a glyph. Hexadecimal (short: *hex*) numbers are written using the digits 0-9 and the letters A-F (usually uppercase). A Unicode codepoint can have 4 to 6 hex digits. Typically, each glyph has one Unicode codepoint. However, fonts may include glyphs with no Unicode codepoint assigned (so-called *unencoded glyphs*) or glyphs with more than one Unicode codepoints assigned (so-called *double-encoded glyphs*).

The glyph Properties panel (**Edit > Properties**) displays the glyphname and the Unicode codepoint of the glyph that is currently active in TypeTool (either selected in the Font Window or opened in a Glyph Window).
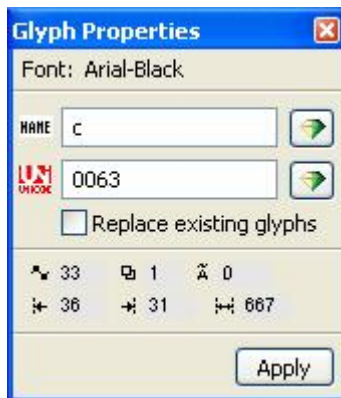
In addition to the Unicode codepoint, each glyph usually has an additional local character code that is depending on the encoding or codepage currently selected in the Font Window.

➘ The rule of thumb is: the encoding of OpenType fonts depends on the Unicode codepoints assigned to the glyphs. The encoding of Type 1 fonts depends on the local character codes of each glyph.

Font Window Modes

The Font Window can be presented in two modes that can be used to browse the font's glyph collection using certain criteria. In some cases, the Font Window mode also influences the encoding of the final generated font -particularly for Type 1 fonts. The two modes of the Font Window are the following:

1. **Names mode.** This mode lists encoding tables. Each encoding table is an ordered list of glyph names and may also include local character codes that correspond to some of the glyphs.

   An encoding table performs one of two functions: it serves as a Type 1 encoding table that is used to determine the character encoding in Type 1 fonts, or it can serve as a glyph arrangement table. The latter is used by type designers to visually arrange glyphs in the design stage, usually of an OpenType font, and is not directly used as the source of the font's encoding (as OpenType fonts are based on Unicode).

   Since an encoding table is based on glyph names, it can reference both encoded glyphs (i.e. those with at least one Unicode codepoint assigned) and unencoded glyphs (those without Unicode codepoints).

2. **Codepages mode.** This mode lists codepages. Each codepage is a mapping of local character codes to Unicode codepoints. The codepage can use one- or two-byte local character codes. Two-byte codepages are used to reference characters in Far-East fonts: Chinese, Japanese, Korean or Traditional Vietnamese.

   A codepage selected in the Codepages mode can be used as the source of encoding of a Type 1 font, or as the source of encoding of a Mac TrueType mapping table, but generally, the selection does not influence the encoding of an OpenType or TrueType font.

   A codepage can only reference encoded glyphs (i.e. those with at least one Unicode codepoint assigned).

Names Mode

To switch the Font window to the Names mode, click on the [NAME] button on the Font Window command bar (top position) or choose the Names mode from the **Mode** context menu (bottom position).

The **Encoding** combo box (top position) or **Encoding** context menu (bottom position) shows the encoding table currently assigned to the font. When you open the combo box/context menu, you will see many encodings that are installed and available in TypeTool. In the bottom position of the Font Window command bar the encodings are shown in groups.

An encoding table performs one of two functions:

- Type 1 encoding table
- glyph arrangement table

As a Type 1 encoding table, an encoding table is used as the source of the character encoding in Type 1 fonts or (in some rare cases) TrueType fonts.

As a glyph arrangement table, type designers use an encoding table to visually arrange glyphs in a particular order during the design process of a font (Type 1, TrueType or OpenType). Such a glyph arrangement table can be used as a "visual map" for the font family so the designer knows what glyphs need to be designed in all members of the family — this way, you will not miss an important glyph.

There is no visual distinction in the Encodings list between Type 1 encoding tables and glyph arrangement tables. Any encoding table can theoretically serve either function. In order, in TypeTool the same user interface element (the encoding tables) serve two different purposes.

An encoding table is either a sequential list of glyph names or maps local character codes to glyph names. TypeTool will look up the glyphs in the current font that have glyph names specified in the encoding table and will present the glyphs visually in the sequence specified by the encoding table. If the encoding table is used as a Type 1 encoding table, the same mapping will be written to the Type 1 font and used as its encoding.

```
41 42 43 20        20 space          !  exclam
5A 5D 6F 55        21 exclam         "  quotedbl
56 57 58 59        22 quotedbl       #  numbersign
5A 5B 5C 5D        23 numbersign     $  dollar
5E 5F 60           24 dollar         %  percent
                   25 percent        A  A
                   41 A              B  B
                   42 B
```

Source text   Encoding table   Font
as sequence
of codes   Identification of characters in the font
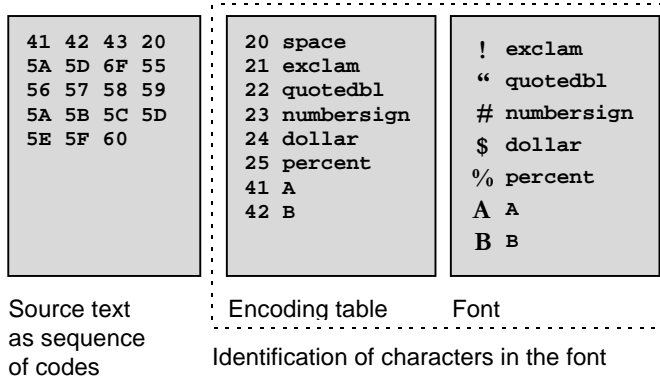
We will now discuss the most common encoding tables included in TypeTool.

Type 1 Encoding Tables

Type 1 encoding tables are used as source for the character encoding in Type 1 fonts.

Type 1 fonts have two fundamentally different kinds of encoding: Standard Encoding and custom encoding.

The rule of thumb is that a Western Roman Type 1 font should be encoded using Standard Encoding, and a non-Western Type 1 font (e.g. Central European, Cyrillic, Greek) should use custom encoding.

With the default settings of TypeTool, if any encoding from the Type 1 Western/Roman group is active in the Font Window, the Type 1 font will be generated using Standard Encoding. If a different encoding is active, the font will be generated using custom encoding that will exactly reflect the active encoding table.

Type 1 Western/Roman group

If **Tools > Options > Generating Type 1 > Encoding Options** is set to **Select encoding automatically** or **Export Unicode codepage…**, then TypeTool will generate a Standard Encoding-encoded Type 1 font if one of the encodings from this group is active. Please refer to the "***TypeTool Options*** (on page 56)" section for more discussion on this. Use any of these encodings when you are working on a typical Western Roman Type 1 font. If you are working on a Western Roman Type 1 font that will be generated as Windows Type 1 and as Mac Type 1, use MacOS Roman as your encoding since this will give you the entire character set that is required to be present in your font.

Adobe Standard Encoding	The "native" representation of the Adobe Standard Encoding
Default Encoding	The simulation of what a Standard Encoding-encoded Type 1 font will appear to the user if installed on the current operating system
MS Windows 1252 Western (ANSI)	The simulation of what a Standard Encoding-encoded Type 1 font will appear to the user if installed on Microsoft Windows
MacOS Roman	The simulation of what a Standard Encoding-encoded Type 1 font will appear to the user if installed on Mac OS. Use this when you are working on a typical Western Roman Type 1 font.

Type 1 non-Western groups

If any of the encodings from these groups is active, the Type 1 font will be generated with custom encoding if **Tools > Options > Generating Type 1 > Encoding options** is set to any value except **Always write Standard Encoding**.

These encodings can be used as the source of encoding for single-codepage non-Western Type 1 fonts, e.g. Mac Cyrillic or Windows Greek. If you are creating a non-Western Type 1 font, make sure to select the appropriate encoding here, and also set the matching character set in **File > Font Info > Encoding and Unicode > Microsoft Character Set** and **Mac script and FOND ID**. Only a combination of the correct encoding in the Font Window and the correct character set setting in Font Info will give you a working non-Western Type 1 font.

Example encodings in this group:

MS Windows 1251 Cyrillic	Encoding for a Windows Cyrillic Type 1 font
MacOS Cyrillic	Encoding for a Mac Cyrillic Type 1 font
Adobe Symbol	Encoding for fonts that include mathematical and symbol characters, defined by Adobe.

"Imported" Encoding

When TypeTool opens a custom-encoded Type 1 font, it will try to match the font's encoding to known custom encodings. If the encoding cannot be matched, "Imported" is shown. The "Imported" encoding will also appear if the user opens a .vfb file that uses an encoding not present on this user's machine.

Glyph Arrangement Tables

Type designers use any encoding table as a glyph arrangement table, i.e. to visually arrange glyphs in a particular order during the design process of a font (Type 1, TrueType or OpenType). Such a glyph arrangement table can be used as a "visual map" for the font family so the designer knows what glyphs need to be designed in all members of the family — this way, you will not miss an important glyph.

For example, if your font contains several glyphs representing the 'A' character, like "A.smcp" (for use with the small caps feature), "A.titl" (for use with the titling alternates feature), "A.swsh" (for use with the swash feature), "A.subs" (for use with the subscript feature), it could be a good idea to have them appear close to each other in the Font Window:
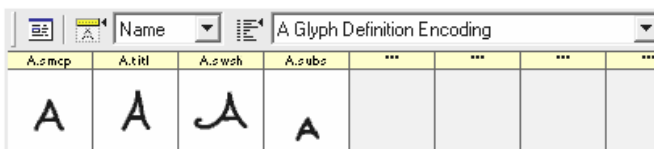
You can easily build a glyph arrangement table yourself — this is explained in the next section.

Remember that if you choose any glyph arrangement table in the Names mode, the glyphs will be displayed in TypeTool in the order that you specified, but the physical arrangement (sequence) of the glyphs in the font is still determined by the glyph indexes.

Also remember that when you create OpenType or TrueType fonts, the encoding table in the Names mode serves as a glyph arrangement table and not as a source of encoding

Normally, OpenType and TrueType are based on Unicode, so the Unicode codepoints that you assign to each glyph are the source of the character encoding. To make sure the encoding of your OpenType or TrueType font is correct, switch to the Codepages mode. You can automatically assign Unicode codepoints to your glyphs by using **Glyph > Generate Unicode**.

When you activate a different encoding in the Font Window, you will see that the characters in the Font Window are rearranged. Some characters will move below "the yellow zone". Remember that only characters that are "in the yellow zone" are covered by the currently selected encoding.

Codepages Mode

Codepages are tables that map local character codes (one byte long) to the Unicode codepoints. Depending on the size of the page, these tables may have 256 or 65,536 records, one for each possible character code. Long codepages are called double-byte codepages and are primarily used to represent codes used in Chinese, Japanese, Korean or Vietnamese languages.

Codepages are necessary because we need to somehow encode text written in different languages in the one-byte code space. So when we have a text file encoded according to some codepage, we use the codepage table to find which characters were used in this text. We may have two different texts with the same code 192 (decimal), but in one case it may mean the Russian 'A' and in the other case it may mean 'À' (Agrave).

Codepages are used not only to identify characters, but also to simplify text sorting, conversion of lowercase to uppercase characters, spell-checking and in many other applications where it is necessary to know which characters are used in the text.

Because the Unicode character identification standard covers most languages it is usually used as the destination information in the codepage tables. Here is an example of fragments from two different codepages that map the same codes to different Unicode codepoints:

MS Windows 1252 Western	MS Windows 1251 Cyrillic
0xC0 0x00C0	0xC0 0x0410
0xC1 0x00C1	0xC1 0x0411
0xC2 0x00C2	0xC2 0x0412
0xC3 0x00C3	0xC3 0x0413
0xC4 0x00C4	0xC4 0x0414
0xC5 0x00C5	0xC5 0x0415
0xC6 0x00C6	0xC6 0x0416
0xC7 0x00C7	0xC7 0x0417
0xC8 0x00C8	0xC8 0x0418
0xC9 0x00C9	0xC9 0x0419
0xCA 0x00CA	0xCA 0x041A
0xCB 0x00CB	0xCB 0x041B
0xCC 0x00CC	0xCC 0x041C

Many different codepages have been defined for many languages and different operating systems. TypeTool includes descriptions for 300+ codepages — all the known Windows, OS/2, MS DOS, Mac OS codepages plus a few others like the Polytonal Greek, Russian KOI-8 and NeXT Step codepages.

In TypeTool a codepage is a filter through which you can "look" at your font to see how it will work in different environments. For example, you might include many Unicode characters in your font and see how it would work if it was installed in OS/2 with the Arabic language selected. This gives you the opportunity to easily create fonts that will be properly encoded and will always work correctly.

To select a codepage in the Font Window:

1. Switch the Font Window to the Codepages mode by clicking on the **Codepages** ▦ button.

2. The encoding selection combo box will show the names of all available codepages:

MS Windows codepages come first, MS DOS codepages follow. All other codepages are sorted according to their names.

Since all codepages are divided into groups they are available in submenus of the **Encoding** menu if the Font window command bar is in bottom location.

3. Select the codepage that you want from the list and you will see the Font Window change. All the characters that are in the codepage appear "in the yellow zone". All other characters are in the "white" area below. Select the MS Windows 1252 Western (ANSI) codepage and you will see how your font will look in the Windows standard (Latin 1) codepage.

All codepages in TypeTool are defined in editable text files, so you can change any codepage if you think it is wrong (please let us know!) or you can define your own codepage. We do not recommend changing any of the codepages supplied with TypeTool. They are extensively tested and are based on the documents from the companies who supply them.

Put your custom codepage definitions (.cpg files) into the [Shared user data folder]\Codepage folder (typically C:\Documents and Settings\Your Username\My Documents\FontLab\Shared\ Codepage) if you want to make the codepages available to all recent Fontlab Ltd. applications, or in the [Application user data folder]\ Codepage folder (typically C:\Documents and Settings\Your Username\My Documents\FontLab\TypeTool3\Codepage) if you want to make the codepages available within TypeTool only. Refer to **Tools > Options > General Options > Folders and paths** for the actual locations of these folders. All custom .cpg files should be located in one of these folders.

Double-byte

If your font contains many glyphs from one of the Far-East languages you may need to use double-byte codepages. If you select one of these codepages, you will see an additional control to the right of the codepage selection list in the toolbar:

or on the Font window bottom command bar:

This control allows you to select a "page" of the codepage. Theoretically, we may have 256 pages of 256 codes each, which give us 65,636 codes. In practice none of the known codepages has that many codes and usually less than half of that number.

Using the Font Window

The glyph chart in the Font window is a visual representation of all the glyphs in the font. To modify the font you have to learn how to use the glyph chart: navigate, select glyphs and select commands.

Navigating

One of the glyphs in the Font window is the "current" glyph. It is specially highlighted:

You can see the current glyph name and its Unicode codepoint in the bottom command bar:

Glyph: A [0041] Selected: 1 / 1030

To view different parts of the font in the Font window you can either use the vertical scroll bar or the auto-scroll mode: if you place the mouse anywhere in the chart; hold down the left mouse button; and move the mouse cursor above the top or bottom of the chart it will scroll up or down accordingly selecting the glyphs.

You can also use the SPACE key to scroll the Font window. Press the SPACE key, hold down the left mouse button and drag the mouse to scroll the window vertically. If you have a wheel on your mouse you can use it to scroll the Font window vertically.

Alternatively you can use the keyboard keys to navigate in the font chart:

Arrow keys	Moves the current glyph highlight one cell right, left, up or down, according to the key used
Ctrl+Right arrow	Moves 2 cells right
Ctrl+Left arrow	Moves 2 cells left
Page Up and Page Down	Moves the glyph highlight one screen up or down
Home	Moves the glyph highlight to the leftmost glyph on the current row
End	Moves the glyph highlight to the rightmost glyph on the current row
Ctrl+Home	Moves the glyph highlight to the first glyph on the chart
Ctrl+End	Moves the glyph highlight to the last glyph on the chart

Selecting

In addition to the current glyph you can select sets of glyphs in the font chart. These selections behave similarly to selected text in a text editor — you can copy selected glyphs to another place in the font or to a different font; you can apply different effects to the selection; etc. Selected glyphs have inverted colors. The last selected glyph is the current glyph:

To select one or more cells, hold down the left mouse button on the first or last cell of your selection, and quickly drag the cursor across the cells you want to select. You will see the selection highlighted. If you drag the cursor outside the visible part of the chart, it will scroll accordingly. To cancel your selection, click on any glyph cell.

Alternative: Using the cursor keys on the keyboard, set the current cell highlight on the first (or last) cell of a selection, then hold down the **SHIFT** key. Move the current cell highlight (as described earlier) to select the cells.

Selection does not have to be continuous. If you hold down the **CTRL** key, you can select and deselect cells in any order and combination.

Context Menu

Most commands available in the Font window can be selected from the context menu.

To open the context menu, click the right mouse button anywhere in the chart area or press the **SPACE** key once.

Here is a sample of the Font window context menu:
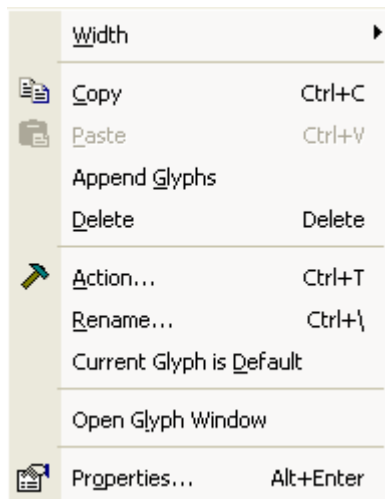
Here is what the commands mean:

Width	Allows you to easily select one of the predefined advance widths of the Font window. The width is defined in cells.
Copy	Copies the selected glyphs onto the Clipboard. Same as the Copy command from the Edit menu
Paste	Places glyphs from the Clipboard into the font starting from the first selected cell. Same as the Paste command from the Edit menu
Append Glyphs	Appends glyphs from the Clipboard to the current font
Delete	Deletes the selected glyphs. Same as the Delete command from the Edit menu
Action	Opens the Actions dialog box. Refer to the "***Actions*** (on page 301)" chapter for more detailed information about actions. Same as the Action command from the Tools menu

Rename	Opens a rename dialog box
Current Glyph is Default	Selects and marks the current glyph as the "default glyph" that is used in Type 1 fonts to represent glyphs that are not present in the font
Open Glyph Window	Creates a new Glyph window and opens the current glyph in it
Properties	Opens the glyph properties panel for the current glyph or selected glyphs.

Rearranging Glyphs

You can change the positions of glyphs in the font chart by moving them to a new place. Note that moving glyphs is an undoable operation.

To move glyphs in the font chart:

1. Select the glyphs that you want to move.

2. Position the mouse cursor on the selected glyphs.

3. Hold down the left mouse button.

4. Drag the glyphs to the new position. Release the button to finish moving.

If you rearrange glyphs by moving them to cells that are already occupied by some existing glyphs, you will see a dialog box prompting you to choose whether to replace the existing glyphs or save them by moving them to the end of the encoding:
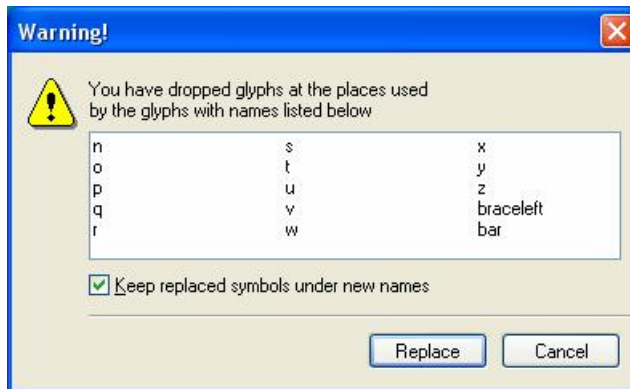
Leave **Keep replaced glyphs under new names** checked to save the glyphs (I.e. put the new glyphs in the cells and move the existing glyphs to cells at the end of the encoding) or clear it to replace them (I.e. delete the existing glyphs).

Note that even if source selection is not continuous the destination selection *will* be continuous:

If you are working in the Codepages or Names mode, when you move glyphs they get new names but keep their old Unicode codepoints. You must assign proper codepoints later.

Saving the Font

Most of the font-modification operations are not undoable, so we recommend you save your work regularly.

To save a font that you have opened from an existing font file (in FontLab format) or imported (from other format), use the **File > Save** command or click on the **Save** 💾 button on the Standard toolbar.

To save all opened fonts click on the **File > Save all** command or this button on the Standard toolbar: 📑.

Font(s) will be saved in FontLab format (.vfb extension) to the folder where the original font was opened.

If this option in the **General Options > Open & Save** page of the Tools > Options dialog box is active:

☑ Create backup files

TypeTool will save the previous version of your font in the same folder as the currently saved .vfb file but will use the .bak file extension instead. If you would like to go back and open the previous (backup) version of your .vfb file, use **File > Open**, navigate to the folder in that you saved your file and type in ***.bak** in the File name field, then press **ENTER**. You will then see the backup file and will be able to open it.

If you are working with a new font or you want to select the destination folder or change the name of the file, use the **File > Save As** command.

Please note that you cannot save fonts with more than **65,535** glyphs. If you try to save a bigger font you will see a warning message that will recommend splitting a font into smaller parts.

After you select **File > Save As** in the menu, you will see the standard File Save dialog box:
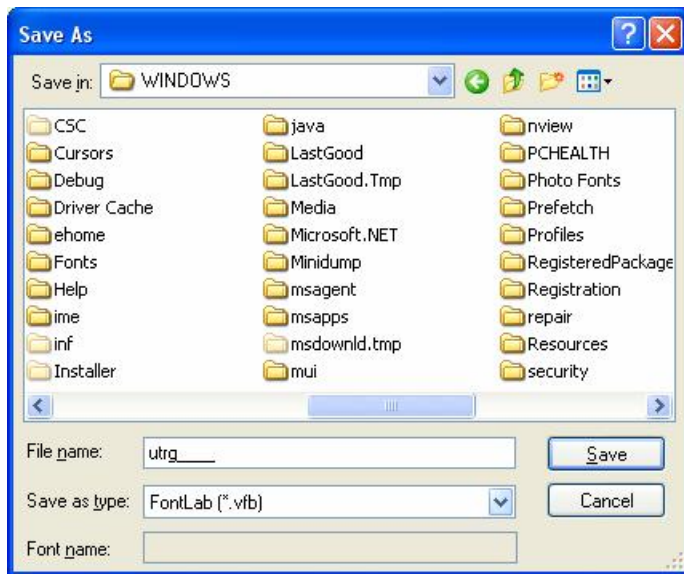
Choose the destination folder, enter the file name and click on **Save** to save your font in FontLab format (.vfb).
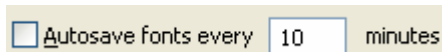
See the "***Generating Fonts*** (on page 349)" chapter to know how to save fonts in other formats.

Autosave

If you want to protect yourself from system or program crashes you can use the **Autosave** function that will periodically save the current font.

To activate and customize this feature, open the Tools > Options dialog box and select the **General Options > Open & Save** page.

You will see the **Autosave** controls:

Use the check box to activate Autosave and enter the time interval (in minutes) at which you want to save the font.

Font will be saved into the Autosave folder within the [Application user data folder], typically My Documents\FontLab\TypeTool3, and will be named using the following structure:

flsX.save.vfb, where fls are the first 3 letters of Font Name and the X is some unique value.

If Autosave was active and you have a system or program crash, you can open your last saved font from the Autosave folder.

Copying and Pasting Glyphs

To copy selected glyphs, select the **Copy** command from the **Edit** menu. Note that this copies not only the glyph outline, but also the glyph information, such as its name. The selected glyphs will be placed in the Windows Clipboard and can be pasted into the same font or into another font by the **Paste** command from the same menu. Glyphs from the Clipboard will be placed starting from the first selected glyph in the destination font. If the destination position is occupied by existing glyphs a warning dialog box appears:
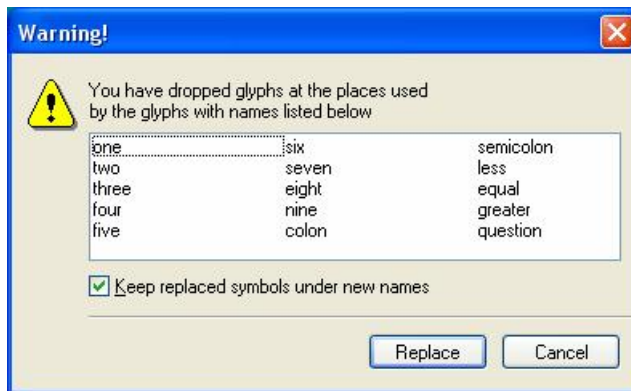
Leave **Keep replaced glyphs under new names** checked to save the glyphs (I.e. put the new glyphs in the cells and move the existing glyphs to cells at the end of the encoding) or clear it to replace them (I.e. delete the existing glyphs).

If you select the **Cut** command instead of the **Copy** command the glyphs will be copied to the Clipboard but will be deleted from the source positions.

If you prefer to use the drag-drop method to copy glyphs within a font window you may do this with the help of the CTRL key. To make a copy of a glyph, select it (you may select many glyphs at once); position the mouse cursor on the selection; click the left mouse button; press the CTRL key; and drag the selection to the place where you want it to be copied. Remember that you need to hold down the CTRL key when you release the left mouse button.

Copying Glyphs to Another Font

You can use two methods to copy glyphs from one font to another:

- Use the **Copy** and **Paste** commands from the **Edit** menu as described, or
- Drag them to the other font and drop them there. The drag-drop method is easier and more visual.

Appending Glyphs to the Font

Instead of the **Edit > Paste** command you can use the **Append** command from the Font window context menu to add glyphs from the Clipboard to the font.

When TypeTool appends glyphs, it respects the glyph names and Unicode codepoints, so on the first attempt glyphs will be placed in the expected code positions in the font.

Here is an example. Your first font contains Latin glyphs but has no Cyrillic glyphs. A second font is a Cyrillic font with the matching style and you want to add Cyrillic support to the first font.

1. Select the Cyrillic glyphs in the second font (this will be easy if you select the 1251-Cyrillic codepage) and copy them to the Clipboard.

2. Return to the first font; right-click on the Font window; and click on the **Append Glyphs** command in the context menu:
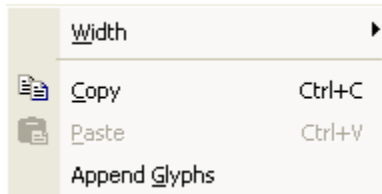
3. The Cyrillic glyphs will be appended to the font with their correct Unicode codepoints and names, so you will not have to re-map the font.

Copying Composite Glyphs

If you copy composite glyphs (instead of having their own outlines composite glyphs are built from references to other glyph outlines) to another font, TypeTool will try to not decompose (replace references to glyph with actual glyph copies) them. Instead it will try to find matching components in the glyph set that was copied or, if some components are not present there — in the destination font.

If TypeTool can completely restore composites in the destination font it will even keep TrueType hinting programs for these glyphs.

Drag-Drop of the Composite Glyphs

If you prefer to use the drag-drop method to copy composite glyphs you have one additional option: when you drop a composite glyph and TypeTool finds that one or more of its components were not selected to copy and do not present in the destination font, it shows a message asking you if you want to copy all the missing components. If your answer is **Copy**, then TypeTool will automatically append all the necessary components to the destination font so that all the composites stay unchanged. Otherwise TypeTool will decompose glyphs.

❱ Note: The described behavior is possible only when both the source and destination fonts have the same Font UPM value.

Duplicating Unicode codepoints

In TypeTool you may assign more than one (up to 63, actually) Unicode codepoints to a glyph. Visually this means that a glyph that has several Unicode codepoints will appear several times when the Codepages mode is selected in the Font window. All copies of the glyph are marked by a small blue mark in the left-top corner of the glyph cells.

To make a duplicate of a glyph, select it (you may select many glyphs at once); position the mouse cursor on the selection; click the left mouse button; press the **CTRL** and **ALT** keys; and drag the selection to the place where you want it to be duplicated. Remember that you need to hold down the **CTRL** and **ALT** keys when you release the left mouse button.

You can later correct Unicode codepoints assigned to the glyph by using the Rename Glyph dialog or the Glyph Properties panel (described later).

Creating New Glyphs

If you want to create a new glyph in an empty place in the font (a grey cell in the Font window), double-click on the cell.

If you are creating glyphs "in the yellow zone", names and Unicode codepoints are assigned to the newly created glyphs according to the selected encoding table.

By default, the newly created glyphs will be blank and will have a default advance width. If possible, TypeTool will also place the grey glyph template image into the bitmap Background layer. You can use it as reference for drawing your glyphs.

Deleting Glyphs

To remove glyphs from the font

1. Select the glyphs that you want to remove.

2. Select the **Delete** command from the **Edit** menu or from the context menu. Or, press the **DELETE** key on the keyboard.

3. A dialog box appears asking you if you are sure that you want to delete.

↘ Note 1: Deleting glyphs from the font is not undoable, so save your work before deleting glyphs.

↘ Note 2: Some glyphs may have a blue mark in the top-left corner — such glyphs are "double-encoded" i.e. have more than one Unicode codepoint assigned. Deleting such glyphs will not physically remove the glyphs; instead, it will remove the selected Unicode codepoint.

Searching for Glyphs

Sometimes you need to find a particular glyph in your font, especially in large fonts. Select the **Find** command in the **Edit** menu or press CTRL+BACKSPACE on the keyboard. You will see a dialog box:
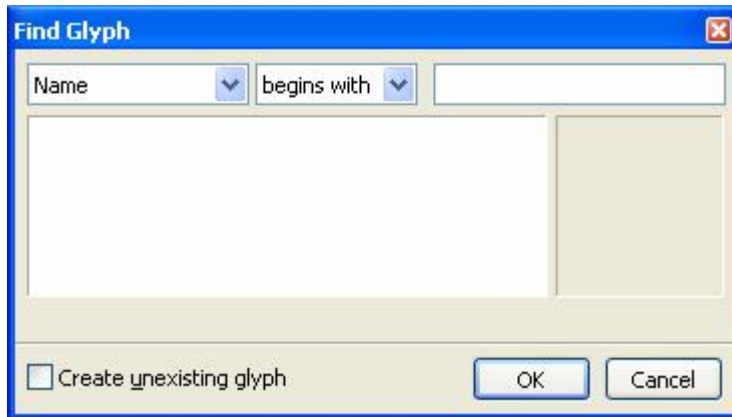
To find a glyph:

1. In the left-top combo box select the method by which you want to search for the glyph:

Name	Searches for the glyphname
Code	Searches for the decimal local character code of the glyph in the current encoding or codepage
ANSI character	Searches for the glyph that is mapped to one of the ANSI glyphs in the selected codepage or encoding
Unicode codepoint	Searches for glyphs with Unicode codepoint attributes given
Width	Searches for glyphs with the advance width in the selected range
Bottom, Top	Searches for glyphs whose bottom or top line falls in the specified range
Components	Searches for glyphs that have the specified number of components
Glyph index	Searches for glyphs with their index attributes given.

127

3. In the combo box to the right of the method select the comparison factor: begins with, equals to, less than, more than, etc.

4. In the right-top editing field enter the information (depending on your selection) that will be used to find the glyph.

5. The names of all the glyphs that match the criterion will appear in the list.
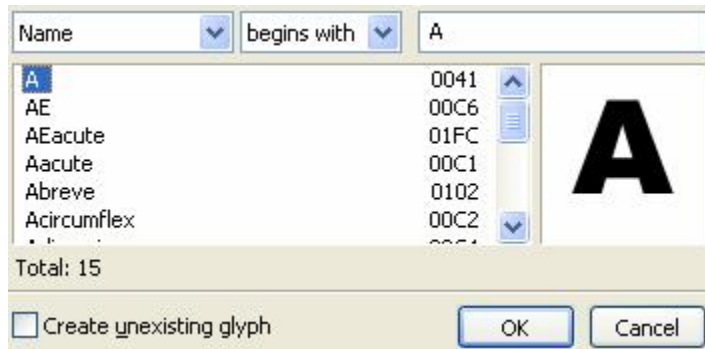
Select the glyph name that you want (its preview appears in the preview panel) and click on **OK**, or enter more information to narrow your search.

Renaming Glyphs

Usually it is not necessary to manually rename glyphs because their names and Unicode codepoints are assigned automatically when you move glyphs in the Font window. But if you want to see the information and correct it, select the **Rename Glyph** command from the **Glyph** menu, or press **CTRL**+\ on the keyboard.

You will see a dialog box:
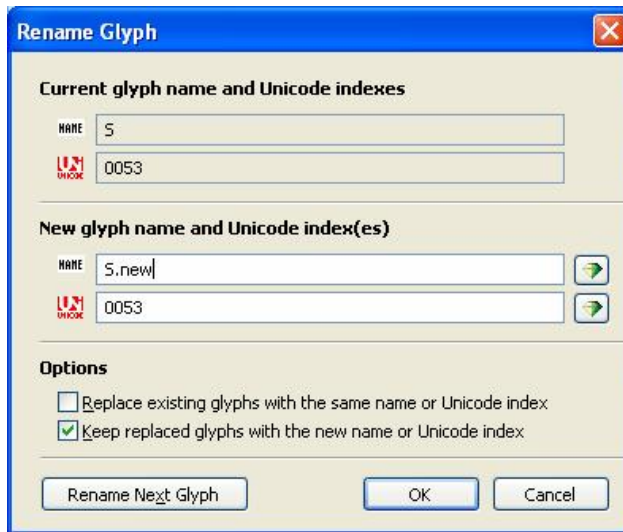
In the top part of the dialog box you see the current name and Unicode codepoint (indexes) of the glyph. In the middle there are two edit fields where you may change the information. Below them lie the options controls.

To change a glyph's name enter a new name in the **Name** field. If this glyph has a properly assigned Unicode codepoint and you want to find the name mapped to that index in TypeTool's database press the **Auto** button to the right of the edit field and TypeTool will fill in the **Name** field for you.

If the option **Replace existing glyphs with the same name or Unicode codepoint** option is not checked then, if you enter a name that is already assigned to one of the font's glyphs, the **OK** button will be disabled and you will not be able to assign that name. Switch the option on to allow TypeTool to replace glyphs. Use the next option to control how TypeTool does the replacement.

Use the **Unicode** edit field to change a glyph's Unicode codepoints. You may enter more than one Unicode codepoint separated by a space. Use the **Auto** button to find the Unicode codepoints mapped to a glyph's name in TypeTool's database.

Press the **OK** button to assign a new name to the glyph. You will see that the glyph moves to a new place in the Font window depending on the currently selected encoding vector or codepage.

If you want to rename more glyphs, press the **Rename Next Glyph** button. A new name will be assigned to the current glyph (as if you had pressed the **OK** button) and data from the next glyph will appear for editing.

Generating Unicode codepoints

To automatically generate Unicode codepoints for all the glyphs in the font, select the **Generate Unicode** command from the **Glyph** menu.

You will see the warning dialog box. Click on **Yes** to completely re-generate Unicode codepoints for all the glyphs. Then TypeTool will:

1. Remove all Unicode.

2. Search the name-Unicode database for each glyph's name.

3. If the name is in the database it adds the Unicode codepoint linked with this name to the glyph's list of Unicode codepoints.

4. Because the database may link more than one Unicode codepoint with a name, steps 2 and 3 are processed whenever a glyph's name is found in the database.

Structure of the Name-Unicode Database

The database that links Unicode codepoints and glyph names is nothing more than a text file, standard.nam or agl.nam, located in the [shared default data folder]\Mapping folder that has the following structure:

```
%%FONTLAB NAMETABLE[: Database_name]
0x0000 .notdef
0x0002 nonmarkingreturn
0x0020 visiblespace
0x0020 space
. . . . . . . . .
```

The first line of this file is a signature that is used to show that this file is a properly defined database file. This line may contain the database name like in agl.nam:

```
%% FONTLAB NAMETABLE: Adobe Glyph List
```

The lines that follow the signature have a very simple structure:

```
<Unicode codepoint> <name>
```

The Unicode codepoint may be in decimal or hex (started with '0x') form. The name should not have any spaces. Names are case sensitive.

One Unicode codepoint may be linked with more than one name and several Unicode codepoints may be linked with one name.

If the name is preceded with the '!', it means that Unicode may be generated from the name but none of the marked names may be generated when the Unicode codepoint is known. This is necessary when none of the glyph's names is included in the list of standard names supported by Adobe (Adobe Glyph List). This feature makes it possible to generate correct Unicode codepoints for incorrectly named glyphs but will never assign incorrect names.

You can extend these files in any text editor, but we strongly recommend not changing them.

Removing Unicode Information

If you want to reset the Unicode information in your font, select all glyph cells and choose the **Clear Unicode** command from the **Glyph** menu. TypeTool will remove the Unicode codepoints from all selected glyphs. This command is not undoable.

The Font Map Panel

When you work with really big Unicode-encoded fonts, you may need to have an overview of your whole font. TypeTool has a special panel, called the Font Map, which can represent the entire Unicode code space as a set of 256 x 256 images where every pixel represents a double-byte code and every image is a plain.

Every pixel row in this image represents a Unicode page — 256 Unicode codepoints which begin with the same code. For example, codes A700-A7FF will form one row.

Every pixel in the row represents an individual code.

To open the Font Map panel, use the **Font Map Panel** command in the **Window** menu. You will see a panel that consists of the code image, toolbar and status bar:
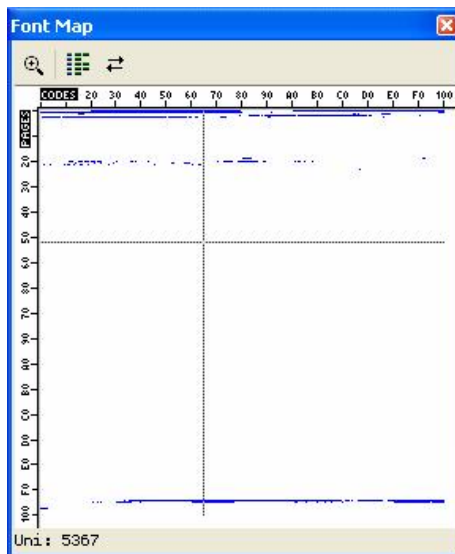
The image represents Plain 0 of the whole Unicode codespace: codes 0000- FFFF.

The buttons on the toolbar mean:

🔍	Turns on zoom mode
▥	Changes the Font Map to double-byte codepage mode
⇄	Updates the contents of the Font Map

By clicking on the 🔍 button on the toolbar you can zoom in on part of the Font Map:
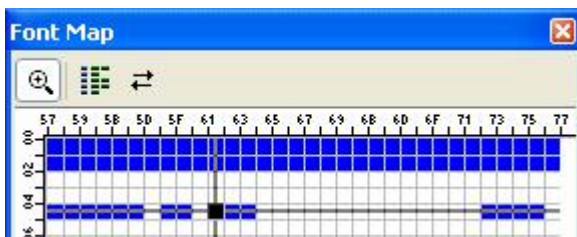
In this mode it is much easier to manage individual codes. To scroll a zoomed Font Map, hold down the left mouse button and drag the cursor beyond the Map borders.

If you click on the Font Map, you will see the current Unicode codepoint appear on the status bar below the Map image. The current code is highlighted with a cross hair.

Double click on any code in the Map to jump to the glyph that is mapped to it.

To switch to another plane of the codespace, use the **Plane** control in the status bar:
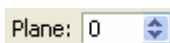
Font Map automatically tracks changes you make to the font. If you are not sure that it is correctly updated click on the ⇄ button to manually update the Map.

Managing Double-Byte Codepages

If you are working on a CJKV (the acronym for Chinese, Japanese, Korean and Vietnamese) font, you may want to look at your font in a double-byte codepage.

Open the Font Map panel and in the Font window of your font select one of the double-byte codepages.

You will see this button ▦ enabled in the Font Map toolbar. Click on it and you will see the Font Map rearrange to represent your font with the applied double-byte codepage. In this mode every row represents 256 glyphs that are "assigned" to the specific first byte.

In the following image you can see a Traditional Chinese font in Unicode mode (in the left image) and in Codepage 950 mode (right):
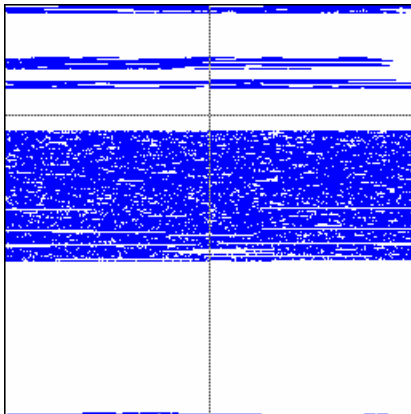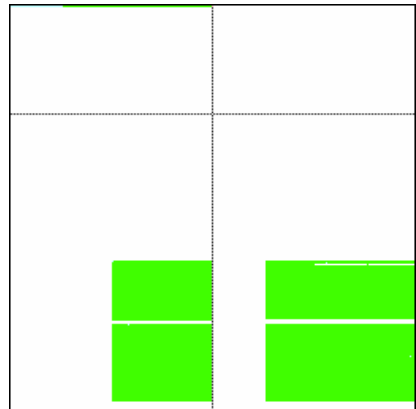
Unicode mode   *Codepage 950*

In the codepage mode green pixels represent codes in the codepage that are covered by one of the glyphs in the font. Cyan pixels mean codes in the codepage that are not covered by any glyphs in the font.

Working with Multiple Fonts

In TypeTool you can open many fonts at once. Since every font has its own Font window sometimes the TypeTool workspace becomes so crowded with windows that finding a particular font is not easy. This section will explain how to use TypeTool tools that are specially designed to help you manage many open fonts simultaneously.

To open many fonts you can use the standard **File > Open** command, then select many font files using **CTRL** and **SHIFT**-click on in the File Open dialog box. You can also select font files in Windows Explorer and drag them to the TypeTool window — all of them will be opened.

When you open 15 fonts the TypeTool window might look like this:
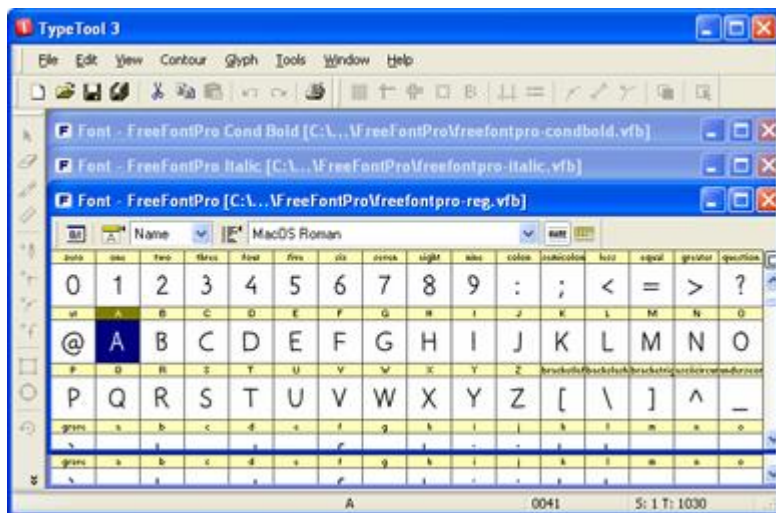
which is not the best way to work. Add to this image a few open Glyph and Metrics windows and you will see why workspace management is necessary.

Windows List

The easiest way to manage open windows is to use the **Window** menu. It contains some very useful commands:

Cascade	Organizes open windows in a cascade like in the image above
Tile horizontally **Tile vertically**	Organizes windows like tiles on a rectangular floor
Windows…	Opens the windows management dialog box.

Choose the **Windows** command and you will see a dialog box:
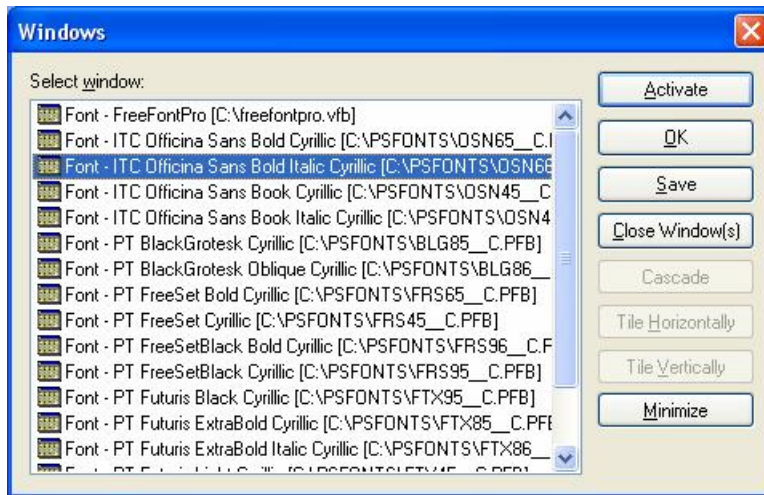
Most of the dialog box is covered by the list of open windows. Select one of the windows in the list and click on the **Activate** button to activate that window and move it to the top.

To close one or more windows, select them in the list and click on the **Close Window(s)** button.

Select two or more windows in the list and click on **Cascade, Tile Horizontally** or **Tile Vertically** to perform one of the operations only with the selected windows. All other windows will be automatically minimized.

Use the **Minimize** button to minimize selected windows.

Applying Modifications

You can find many modification commands in the **Contour, Glyph** and **Tools** menus that can be applied to the open glyph in the Glyph window (see the "*Glyph Window* (on page 139)" chapter), but most of them are applicable to the glyph(s) selected in the Font window.

To apply a modification command, select the glyphs in the Font window and choose the appropriate command in the **Contour, Tools** or **Glyph** menu. For example, to convert glyphs from TrueType outlines to Type 1 outlines, select the glyphs that you want to convert and choose the **Contour > Convert > Curves To PostScript** command. If more than 128 glyphs were selected for transformation, you will see the warning message:

Attention!	×
⚠ You will apply selected action to the 1203 glyphs. This operation is not undoable. Do you want to continue and transform glyphs?	
	[Yes] [No]

Click on **Yes** and TypeTool will apply the command to all selected glyphs.

You can find descriptions of all modification commands in the chapter "*Glyph Window* (on page 139)". Please also refer to the "*Actions* (on page 301)" chapter for related information.

The Glyph Window

The Glyph Window is a standard tool in all FontLab-based applications. It is a universal and very powerful contour-editing module that also allows you to perform many font-specific operations.

Glyph Window Contents

Open the Glyph Window by double-clicking on any glyph sample in the Font Window or Metrics Window.
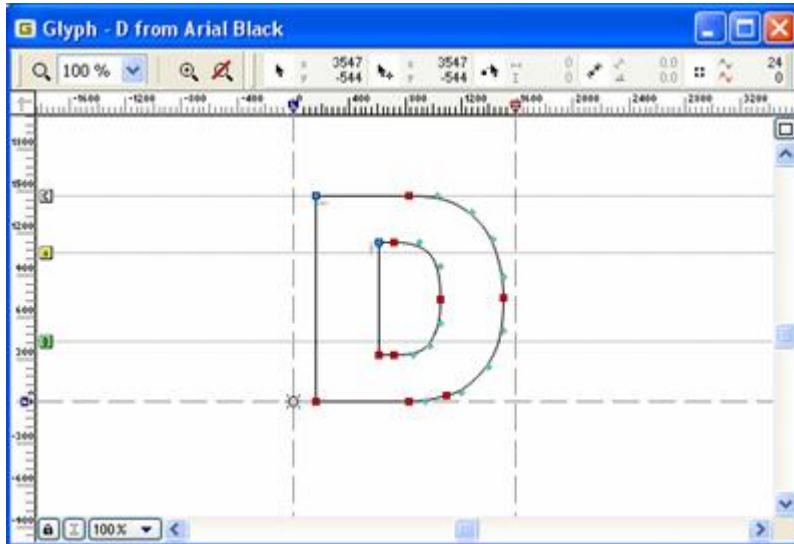
The Glyph Window

The Glyph Window has the following parts:

- Toolbar area
- Editing Field
- Top and left rulers
- Left-Top box
- Scroll Bars
- Lock button
- Meter panel button

The *Local Toolbar* is the command center of the Glyph Window. There is a combo box with zoom selection and two buttons that are used to select the zoom level:
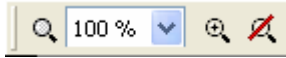
You can drag the local toolbar to any place on your screen or you can dock it at the top or bottom area of the Glyph Window. To show and hide the local toolbar, use this button in the top-right area of the Glyph window:

Below the default (top) location of the Local toolbar and at the left of the window you may see rulers that are used to preview positions of various structures in the glyph space. You may switch the rulers on and off with the **View > Rulers** command or using the context menu that appears if you right-click one of the rulers.

Selecting a Glyph for Editing

You can open a glyph in the Glyph Window using any of the following methods:

- Double click on the glyph's cell in the Font Window to open it.

  If you already have an open Glyph Window with a glyph from the same font, the new glyph will be opened in the same Glyph Window (where the previous glyph was shown). Hold down the **CTRL** key when you double-click on the glyph cell to open it in a separate Glyph Window. Note that if this method does not work it usually means that it is switched off in the Font Window's options in the Tools > Options dialog box.

  You can force TypeTool to always open a glyph for editing in a new Glyph Window. Use the **Double-click on opens a new window** option in the Options > Font Window page to activate this feature.
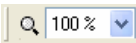
- Click the right mouse button in the Font Window and select the **Open Glyph Window** command to open the glyph in a separate Glyph Window.
- The lock button 🔒 allows direct keyboard access to glyphs in your font. If the button shows a closed lock, single-letter keystrokes are used for keyboard shortcuts such as **Z** for zoom in. If the button shows an open lock, pressing single keys on the keyboard takes you to the corresponding glyph in your font; so pressing **Z** will open the "z" glyph for editing (**SHIFT+Z** will open the uppercase "Z"). To access an extended character, quickly type its glyph name. For example, to open the "ä" glyph, type **ADIERESIS** on your keyboard. Note that usually, just typing **ADI** will do if there is only one glyph in the font with the name that starts with that string. Use **SHIFT** for uppercase names.
- Click the left mouse button on the glyph selected in the Font Window and drag it into any Glyph Window.
- Select the **Find** command in the **Edit** menu and find the glyph that you want to open.
- If you have a wheel on your mouse, hold down the **CTRL** key and scroll the wheel to move to the previous or next glyph.

Creating Glyphs

If you want to create a new glyph in an empty place in the font (a gray cell in the Font window), double-click on the cell. Refer to the "***Creating New Glyphs*** (on page 125)" section in the "***Editing Fonts*** (on page 79)" chapter for details.

Changing the View in the Glyph Window

Use the zoom level and scroll bars to change the view in the editing field of the Glyph Window. By using the scroll bars you can scroll the viewing field of a glyph. With the zoom level you can define how the glyph unit coordinates are converted to screen coordinates and vice versa. If you choose a higher zoom level you will see more details of the glyph and you can do the editing operations more precisely. However, in the larger zoom levels only part of the glyph will be visible so you will have to use the scroll bars to see the different parts of the glyph.

There are fixed zoom levels and custom zoom levels. You can select one of the fixed zoom levels in the **Zoom** combo box located in the upper part of the Glyph Window: . When you choose a fixed zoom level TypeTool will return to this glyph mode on every **Zoom Out** command (or when you press **CTRL+O**).

You can also use the **Zoom** menu in the bottom of the window, it will show the same selection of the zoom levels and is very useful when the Zoom toolbar is hidden:
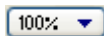
To magnify part of the glyph, select the Zoom tool (button on the toolbar, the + (plus) key or **CTRL+SPACE** on the keyboard) and declare a custom zoom level using a marquee. This zoom level is temporary and you can always return to the previously selected fixed zoom level by clicking on the button (or by clicking on the – (minus/hyphen) key or **CTRL+O**).

Alternative keyboard shortcuts are:

Ctrl+Space	Zoom in
Ctrl+Alt+Space	Zoom out
Ctrl+1,2,3,4,5,6	Set fixed zoom mode from 6,25% to 200%
Space and drag	Scroll (hand cursor appears)

After you select the zoom tool, move the mouse pointer to one of the corners of the rectangular area that you want to zoom on and click the left mouse button. Then, holding down the left mouse button, define the zoom-in area by dragging the cursor to form a rectangle. Release the left mouse button and the new custom zoom level will be selected.

If your mouse has a wheel, use it to scroll the Glyph Window vertically, hold down the SHIFT key to scroll it horizontally, hold down the ALT key to zoom in and out, and hold down the CTRL key to go to the next or previous glyph.

Quick Zoom Selection

You can quickly change the zoom level of the Glyph Window by selecting the **Zoom In** or **Zoom Out** command from the **View** menu. Alternatively you can press the **Z** key for zooming in or the **X** key for zooming out.

This command increases or decreases the zoom level by a factor of two. If the mouse cursor is in the editing area of the Glyph Window the new zoom level will be centered around the cursor position.

These keys are active even when you drag something with one of the editing tools.

If you have a wheel on your mouse, you can hold down the **ALT** key and use the mouse wheel to change the zoom level.

Vertical Alignment Options

When you select 100% as the zoom value, TypeTool needs to choose a scaling factor to fit the font unit space in the Glyph Window. Two vertical levels in the font space define this scaling: Visual Ascender and Visual Descender:
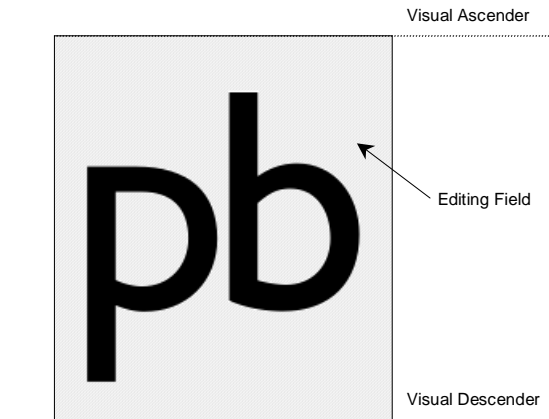
When you select 100% zoom, it means that Visual Ascender is fitted to the top of the editing field and Visual Descender to the bottom.

The same values are used to build the icons that you see in the Font Window.

To set Visual metrics, use the **Glyph Window > Dimensions** page of the **Tools > Options** dialog box:
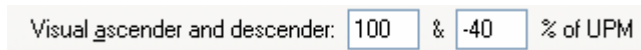
Values are measured in percentage of the font UPM size, so -20% is -200 if UPM size is 1000 and -410 if UPM size is 2048.

147

Tools and Operations

TypeTool's Glyph Window may work in several modes. The three most important modes are:

⬉	**Edit mode**	The main mode used to draw new glyphs, move everything in the glyph, from guidelines to nodes to glyph margins
✎	**VectorPaint mode**	A set of tools used to create new glyphs or modify existing glyphs using vector drawing tools that simulate natural bitmap tools
✎	**Meter mode**	Used to measure contours, distances or angles.

Other modes include two additional operations:

⊞	**Free Transform**	Scales, rotates or skews the selected portion of the outline
⊟	**Move and Scale Background**	Sets the size and position of the bitmap background layer

Use the buttons on the Tools and other toolbars or keyboard shortcuts to switch modes. The three most important modes are:

ALT+**1**	Edit mode
ALT+**3**	VectorPaint mode
ALT+**4**	Meter mode

Though the Tools toolbar contains a button only for Meter mode you can easily customize it (**Tools > Customize**) or even create the custom special toolbar containing buttons for switching modes:

Edit Mode

he Edit mode is the most important in TypeTool. In this mode you can modify the contents of all the editing layers.

All operations performed with the edit tool can be undone with the **Undo** command of the **Edit** menu, or by clicking on the **Undo** button 🔄 on the toolbar at the top of the Glyph Window. You can undo up to 200 operations. All undone operations can be redone with the **Redo** command of the **Edit** menu or with the **Redo** button 🔁 on the toolbar.

In Edit mode you can use eleven different Edit Tools. You can easily choose one of the tools using the Tools toolbar:
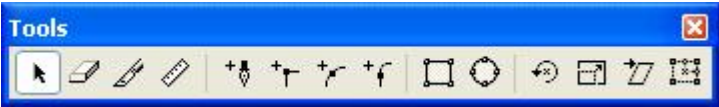
Note that the Tools toolbar also contains buttons of other modes.

Alternatively you may use the keys from **1** to **9** to quickly select edit tools:

▸	1	**Edit**	Main tool, used to drag objects on the editing layers and perform other operations. In the following chapters we will assume that this tool is active in the Edit mode
◿	2	**Eraser**	This tool is used to quickly remove unnecessary nodes
◿	3	**Knife**	Tool to insert nodes and break outlines
✸	4	**Magic Wand**	Tool to quickly select contours (click anywhere near the contour and it is selected). Note that this tool is not available in the Tools toolbar by default

	5,	**Add Corner,**	Tools to create new contours or insert nodes
	6,	**Add Curve,**	
	7	**Add Tangent**	

	8	**Bézier Drawing**	Tool to draw the contour with the Bézier curves

		Rotate,	Tools to quickly transform outlines.
		Scale,	
		Slant	

Temporary Activating the Edit Tool

To temporary activate the Edit tool while you are using any other tool press and hold down the CTRL key.

Snap-to Distance

In the following sections we will discuss how to use the Edit tools to modify the outline and other editing layers. All other tools will be explicitly named.

When you need to select a node or any other object on any of the layers, you need to click on it with the mouse. You do not need to click on the object precisely, but you must be within a certain distance, which is called the "snap-to distance".

Snap-to zone around a node
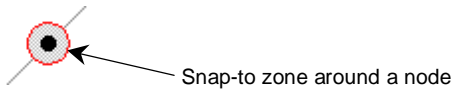
Snap-to is used when you select an object for which the feature is allowed. By default the snap-to distance is set to 3 screen pixels, but you can change it on the **Glyph Window > Dimensions** page of the Options dialog box:

Editing Layers

In TypeTool every glyph contains several editing layers. Some of them are used when the font is exported; others are TypeTool-only and are used to help you work with the glyph. Below is the list of all the layers that you can see in the Glyph Window. Later we will describe them in full detail.

	Outline	Main layer containing the glyph's outline
	Grid	Regular grid which helps to align the outline
	Guidelines	Horizontal, vertical and/or diagonal guidelines
	Hints	Type 1 hints — pairs of vertical or horizontal lines set at a fixed distance
	Mask	Outline template
	Background	Bitmap background
	Glyph metrics	Glyph metrics — left and right sidebearings and a baseline
	Vertical metrics	Vertical font metrics, such as ascender, descender or cap height
	Global Mask	Global (font-wide) mask

You can control the layers' appearance and features with the **View** menu:

Show layers	Lists all layers (and a few other options) and lets you switch them on or off. You cannot switch off (hide) the Outline layer
Lock layers	Lists the layers that you can lock to protect from accidental modification. E.g. if the Outline layer is locked you will not be able to add, move or delete any nodes or change any curves
Snap to layers	Controls which layers have the "snap to" feature activated.

If you frequently need to switch some layers on and off you can drag the
View > Show Layers menu and convert it to a toolbar:
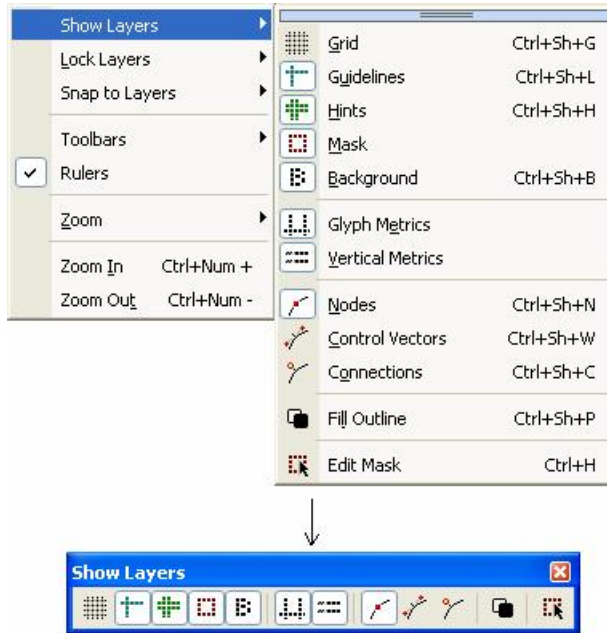
The same operation is possible with the **View > Lock Layers** menu:

In the following sections we will describe all the editing layers and their
modification and control.

153

Outline Layer

The Outline layer is the most important of all the layers. It stores information about the glyph shape while all the other TypeTool editing layers and most of the tools are designed to help you create good outlines. Before we turn to the outline editing tools let's talk about outline structure.

Units of Measurement

All glyph drawings in a font are measured using a standard coordinate system. One unit of this system is called a font unit. Any point on a glyph outline has x and y coordinates, measured in font units. Other distances such as advance widths, kerning values, component positions etc. are also measured in font units.

All glyphs in a font have a common reference scale that defines the relation between the glyph drawings and the point size of the font. This reference scale is called *UPM size (Units Per eM size)*. Whenever the font is displayed at a certain point size, it is sized so that the UPM size matches the point size. The UPM size of most currently available fonts equals 1,000 units. This means that whenever the font needs to be displayed at 10 pt type, the 1,000 units are scaled to 10 typographic points, i.e. to 10/72 of an inch. You can define the UPM size for your font on the ***Font Info > Metrics and Dimensions*** (on page 330) page.

In typical fonts, the UPM size is 1,000 units and that the uppercase letter "H" is around 700 units high. If you set some text in your font at 10 pt size, the 10 pt will correspond to the UPM size, that is 1,000 units. This means that your uppercase letter "H" will be 7 pt high. In a different font, the uppercase letter "H" can be 800 units high so set at 10 pt, the letter "H" will be 8 pt high. This explains why point size of typeset text is not really related to any "graphic" element of the typeface, and why different typefaces appear visually larger or smaller when typeset at the same point size.

If you would like to make your letter "H" to appear visually larger when set at a specific point size, you need to increase the ratio of the size of the letter in units to the UPM size. With the letter "H" being `700` units high and the font having the UPM size of `1,000`, the ratio is `700/1,000 = 0.7` or `70%`. To increase this ratio, you have two possibilities: you can either increase the size of the letter "H" in units (i.e. rescale the glyph) or you can reduce the UPM size without rescaling the glyphs. The visual effect will be identical. Let's say you wish to visually increase your letterforms so the letter "H" is `8` pt high (rather than `7` pt) when the font size is `10` pt. Your desired ratio of the height of the letter "H" to the UPM size will be `0.8`. So you can either *increase* the size of the letter "H" so it is `800` units high and *keep* the UPM size of `1,000`, or you can *keep* the height of the letter "H" at `700` units, but reduce the UPM size to `875`, since `800/1,000 = 700/875`.

Font units are integer numbers — this means that an outline point (a node) can have the `x` coordinate of `334` or `335` but not `334.5` or `334¼`. This means that unlike Bézier drawings in Illustrator or Freehand, glyph drawings in fonts have a finite precision. In a `1,000` UPM font, some extremely thin hairlines or extremely precise details cannot be accurately represented.

The UPM size of `1,000` has been generally accepted as one that offers the best compromise between detail fidelity and memory space efficiency. Practically all Type 1 fonts use the UPM size of `1,000`. However, other UPM sizes are permitted. For example, some font vendors produce OpenType TT fonts with the UPM size of `2,048` while others use `1,000`. Most OpenType PS fonts use the UPM size of `1,000`, though there are a few that use different UPM sizes. To increase the precision of your drawings, you may choose a higher UPM size such as `1,500`, `2,000`, `2,048`, `3,000` or `4,000` or even `1,877`. The maximum UPM size permitted by the OpenType and TrueType specifications is `16,384.`, and TypeTool currently has an internal implementation limit for glyph coordinates of `±32,767`. However, Adobe specifications require that glyph coordinates and advance widths must not exceed `±4,095` in either `x` and `y` directions, regardless of the UPM size.

Note that since the UPM size does not relate to any physical drawing in the font, it is actually impossible to precisely measure the point size of printed text. While you can use a ruler to measure the width or height of some letters, there is no way to reliably tell the point size that was used, because a `700`-units high "H" in a `1,000` UPM font set at `10` pt is physically identical to a `700`-units high "H" in a `1,200` UPM font set at `12` pt, and is also identical to a `1,792`-units high "H" in a `2,048` UPM font set at `8` pt.

Another relevant font parameter that should not be confused with UPM size is the *font height*. The font height is measured in font units and is used to determine the *default linespacing* of text set in a font.
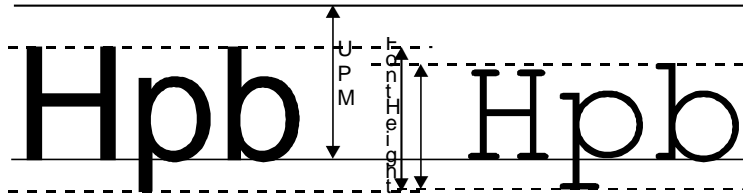
The font height is the distance between the font's most imporant *vertical font metrics*: the *Ascender* line and the *Descender* line. Both lines are defined by the type designer. The Ascender line is typically equivalent to the height of tall lowercase letters such as "b", and the Descender line is typically equivalent to the depth of lowercase letters that extend below the baseline, such as "p". However, the Ascender and Descender lines do not have to be precisely identical with the size of those letters. To ensure equal linespacing when changing fonts within a font family, the Ascender and Descender lines should be equal across all fonts in the family, even though bolder styles typically have slightly shorter lowercase letters than lighter styles. A font may contain glyphs that can extend above the Ascender line or below the Descender line, such as the integral sign. However, such extra-tall glyphs may be clipped when they are displayed on screen (in some applications) or printed (on some printers).

Type 1 fonts have only one Ascender and one Descender line. TrueType and OpenType fonts have three different Ascender lines and three different Descender lines, so setting these values must be done with care — this will be described later.

Reference Points

By default all coordinates are measured relative to the *zero point* of the glyph. This is located at intersection of the baseline and the left sidebearing line:

Reference point

Zero point
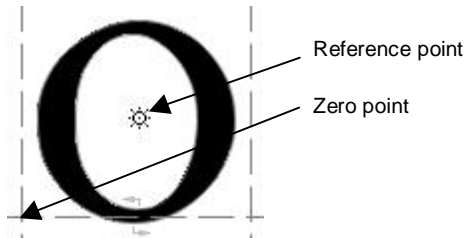
As an alternative, distances may be measured relative to the *reference point*, which may be positioned by the Edit tool to any point in the glyph space. Often a reference point is very useful when you are working on a symmetrical shape.

To set the precise position of the reference point, CTRL-click on it. You will see the reference point properties dialog box and will be able to enter the horizontal and vertical position of the reference point.

By default the reference point is located on the position of the zero point.

Contours

The most important and most complex information in a font is the glyph's shape. All glyphs are defined as a series of contours. All contours consist of a series of segments: straight lines and curves. Nodes — i.e. outline points — define all segments.

Open and Closed Contours

Contours may be open or closed:

All known font formats require contours to be closed, but during outline editing it may be useful to have some contours in an open form and later connect them to each other to build final closed contours.

In TypeTool it is very easy to open closed contours or to close open contours.

Filled and Unfilled Contours

Contours can be of two types: black or white. They can also be of two directions: clockwise or counterclockwise. The basic rule that applies to Type 1 fonts is simple: clockwise-directed contours are white and counterclockwise contours are black. A simpler form of the rule, known as the rule of the *left hand*, is: if you face along the direction of a contour, black (fill) will be on your left side.

Directed contour

Filled interior zone
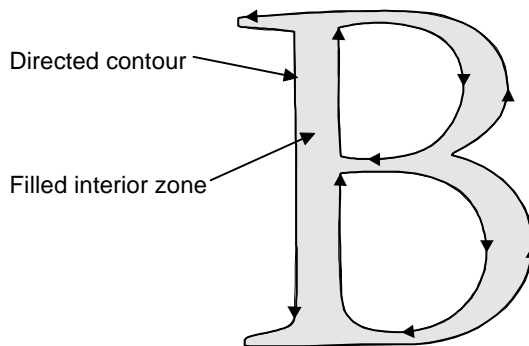
In the TrueType specification the opposite is the case, so a contour is filled on the right hand side. However, not all TrueType rasterizers require glyphs to follow this rule, so it is recommended, but not necessarily required, that you reverse contour directions when you are converting Type 1 fonts to TrueType.

Startpoint and Closepath

All contours have a *startpoint* **(start point).** The startpoint is the first node of the contour. The last node of the closed contour is automatically connected to the startpoint with a straight line, which is called *closepath*. The color of the startpoint in the Glyph Window is blue.

A contour-direction mark always appear on a startpoint:

Curves and Lines

Segments are of three types: straight segments, PostScript curve segments or TrueType curve segments. Straight segments (sometimes called *vectors*) are straight lines that connect two sequential nodes. PostScript curves (also called Type 1 curves) are *Bézier curves* (3rd order, cubic B-splines). To modify the form of the curves two additional sub-nodes are used:
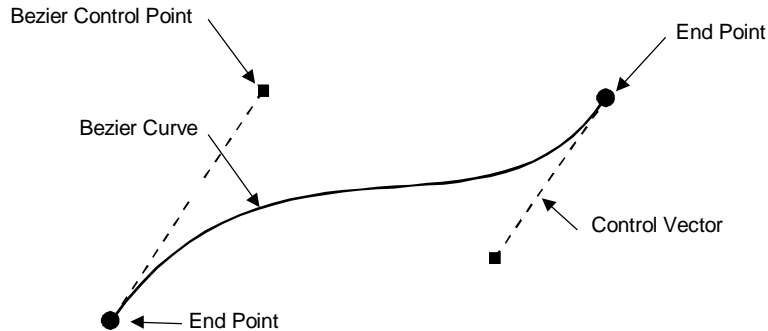
These *sub-nodes* are called Bézier *control points (BCPs)* and the vectors that connect the control points with the curve's ends are called *control vectors*. In the Glyph Window straight segments end in square (in black-white mode) or red (in color mode) dots and curves end with round or green dots. Note that the contour direction plays a role here: it is always the shape or color of the *final* node of a segment that tells you about the type of the segment.

TrueType curves are 2nd-order curves (quadratic B-splines) that have one control point, called the "*off-curve*" point:
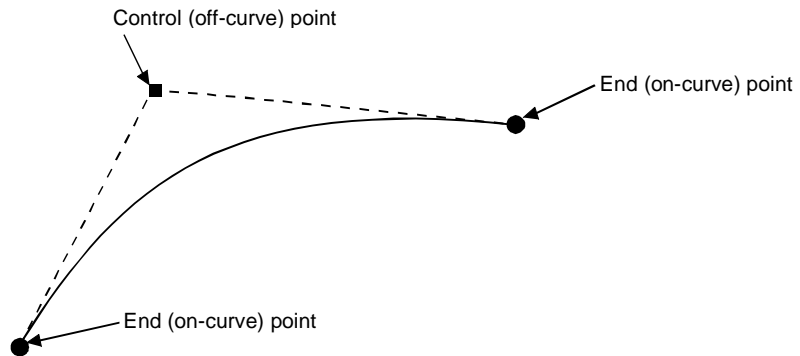
Some TrueType curves may appear linked together and form a long curve with off-curve points only. In such curves, the intermediate on-curve points do not exist explicitly, but are implied by the rasterizer:
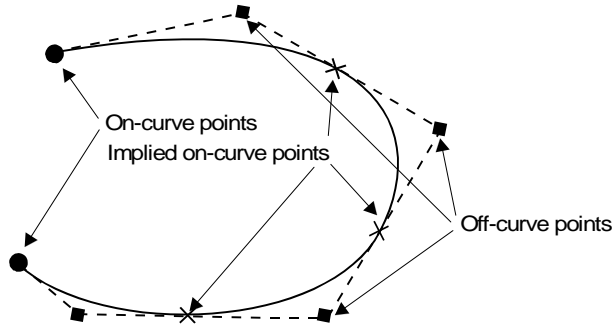
TrueType curves end with points that look exactly like straight segment points. Off-curve points of the TrueType curves have a "plus" (in black-white mode) or light-blue (in color mode) appearance.

Connections

The type of connection between segments is very important if you want to keep the contour smooth at appropriate nodes. There are two types of connections: sharp and smooth.

At a sharp connection, the two connected segments (curve and curve or straight segment and curve) are absolutely free in their angle relative to each other at the connecting node.

At a smooth connection, the direction of the straight segment and the control vector of a curve or the control vectors of two sequential curves are kept collinear (lie on the same straight line). I.e. the angle between the two segments at the node is fixed at 180 degrees.

Smooth connection
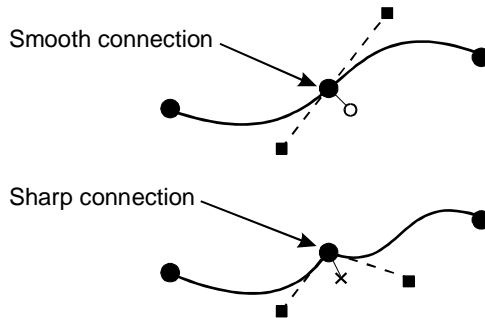
Sharp connection

It is very important to maintain the smoothness of the glyph's contours at the appropriate places. Small corners (sharp connections that are invisible when glyphs are small) become visible (and ugly) when you print large text. Furthermore, rasterizing programs that convert outline glyphs into bitmap images on paper do not like outlines where sharp connections are present in places where the outline should be smooth.

➘   Note: A quick way to change the connection type is to double-click on a node. More detailed control is available on the Properties panel. Use **Contour > Correct Connections** to automatically fix incorrect sharp connections, i.e. sharp connections that can be turned into smooth connections without any change in the shape of the contour.

Node Type

TypeTool has several types of nodes that are represented by different node symbols. The node symbol unifies two essential kinds of information: the type of segments that the node connects (straight segment or curve segment) and the type of connection (sharp or smooth).

	Curve node. (Green) round node symbol indicates a smooth connection between two curve segments.
	Tangent node. (Violet) triangular node symbol icon indicates a smooth connection between a curve segment and a straight segment.
	Corner node. (Red) square node symbol indicates a sharp connection between any types of segments.

A blue node indicates the startpoint. To display the nodes similar to *Fontographer* (http://www.fontlab.com/fontographer/), in black-and-white only, enable **Tools > Options > Glyph window > Appearance > Black/white**. To display the additional color information as in older TypeTool versions, disable that option.

➘ Note: Corner nodes may exist between any types of segments (straight or curve). If possible, you should convert a corner node between two curve segments into a curve node, and a corner node between a curve segment and a straight segment into a tangent node (see the section "*Connections* (on page 162)" above). A smooth connection between two straight segments does not exist — it is always a sharp connection; also, it usually constitutes a "collinear vector" and should be simplified into just one segment.

To visually emphasize the connection type (smooth or sharp) of all nodes, you can enable the Connections layer: **View > Show Layers > Connections**. If enabled, additional connection symbols appear next to each node. A small "x" next to the node indicates a sharp connection; a small "o" next to the node indicates a smooth connection.

Outline Appearance

You can view an outline in contoured or filled mode. These modes are equivalent for all editing operations, but the filled mode is a little bit slower. However, in the filled mode you always see how the glyph will look in the resulting font. Switch between modes with the ⬛ button on the **Show layers** toolbar or with the **View > Show Layers > Fill Outline** command.
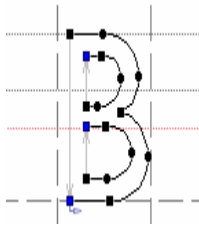
Outline mode

Fill Outline mode

Smoothed Contour

By default a contour is rendered with black color and sometimes this may result in jaggies:
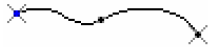
Optionally you can smooth the contour appearance on screen, which will result in a much smoother outline appearance:

To smooth contours, use the **Glyph Window** page of the Tools > Options dialog box**:**
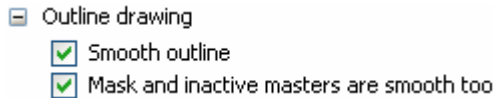
You can use the **Apply** button in the bottom-right of the Options dialog box to check the result of the changes you make in the Glyph Window options.

↘ Note: If your computer is slow and a contour is complex, smoothing the outlines may degrade the performance of the editing tools. Turn it off in this case.

High-quality Preview

No matter which mode is active you can quickly view a high-quality preview of the outline by pressing the '`' key on the keyboard (the key under **Esc**). Until you release the key you will see a high-quality preview of the outline. Note that you can use the  ',' and '.' keys to browse glyphs without releasing the '`' key.

↘ Note: On the U.S. English keyboard layout, the '`' key is located between the **Tab** and the **Esc** keys. On other keyboard layouts, you may need to use a different key, e.g. **Ö** on the German keyboard.

Outline Preview Options

You can choose other options in the **View > Show Layers** menu for previewing the contour layer:

✏	**Nodes**	To show nodes or not
✏	**Control vectors**	To show curve control vectors or not
✏	**Connection mode**	To show connection mode marks

A few more notes about outline appearance:

Selected parts of an outline appear red in color. Selected nodes are marked as red rectangles and they are visible even if non-selected nodes are hidden.

Some options related to outline appearance can be customized in the Appearance and Outline Drawing sections of the Glyph Window page of the Tools > Options dialog box described in the "***TypeTool Options*** (on page 56)" section.

Here is a list of some available options and a description of the features they control:

Small nodes	Nodes may be small or large:

Black/white nodes	Nodes may be colored or black/white:
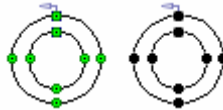
Show node position	One node may be selected as the current node. It will be highlighted and its position will appear on screen:

(446, 211)

To deselect the node, click anywhere in the empty space of the editing field or press the **Esc** key

Smooth outline	Allows one to select between standard and smoothed rendering of the outline:

Leave echo while editing	When editing contours the original contours shape/position is shown gray:

167

Moving Nodes

The most important editing operation is the modification of the contours that build each glyph. You can modify contours in three ways: moving nodes, editing segments using non-node editing, and selecting several nodes and moving them together.

To move individual nodes:

1. If nodes are hidden, make the node that you want to edit visible: switch nodes on with the **View > Show Layers > Nodes** command or click on near the node to make it visible. If you missed and an incorrect node is highlighted, press the **PAGE DOWN** and **PAGE UP** keys to move to the correct node:

2. Drag the node to the new place. It will stick to the objects in other layers if they are visible and snap-to those layers was activated (**View > Snap to Layers** menu).

    Hold down the **SHIFT** key to constrain the direction of the node's movement in 45-degree increments and to snap the cursor to the original node's position.

Options

If you are moving a node that is connecting two Bézier (PostScript) curves you have the following options:

1. If the connection of the curves is smooth, hold down the **SHIFT** key *before click*ing on the node to constrain movement to a line between the curves' control points:
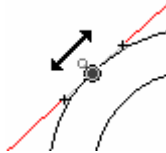
2. If the connection is sharp, hold down the **ALT** key at any time *while* dragging the node to move it without the adjacent control points:
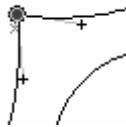
3. If the connection is smooth, hold down the **ALT** key *before* moving the connecting node to keep the connection's curvature optimised. Hold down the **CTRL** key to involve all 4 control points in the process:
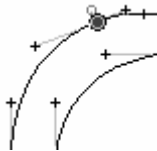
4. When you are editing control points of a Bézier curve hold down the **SHIFT** key *before click*ing on the button to keep the direction of the control vector unchanged.
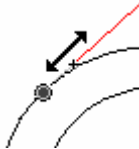
5. If you are moving a control point of a curve with a sharp connection, hold down the **ALT** key to temporarily change the connection type to smooth, so that the adjacent control vector will be collinear

Do not forget that you can hold down the '`' key (the key under **ESC**) at any time to get an instant high-quality preview of the glyph outline as it will print:
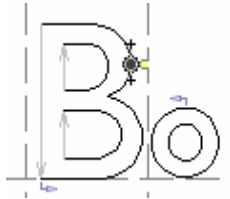
Normal outline   *High-quality preview*

Outline Echo

If you want to see how the outline looked before you moved a node, switch on the Echo mode. Open the Glyph Window page of the Options dialog box and switch on this option:

This is how editing field will look when echo mode is on:
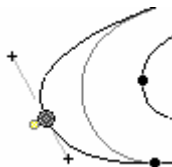
Using the Keyboard

You can use the keyboard to move nodes and to select a node for editing:

Arrow keys	Every keystroke moves current node by one font unit
Shift+Arrow keys	Every keystroke moves a node by 10 font units
Ctrl**+Arrow keys**	Every keystroke moves a node by 100 font units
Page Up	Selects the previous node for editing
Page Down	Selects the next node for editing
Tab	Alternates between the node and Bézier control vectors
Esc	Drops the selection of the current node.

↘ Note: You can make a line or a curve a current object and arrow operations will move it as a whole. Click on a curve or a line with the Edit tool and it will be highlighted by a pair of short lines:

Non-node editing

Sometimes you may want to modify a contour in a more flexible way than by moving nodes. For example, to adjust the shape of a curve in node editing you would usually make the control points of a curve visible and move them to modify the curve. A more intuitive way would be to "grab" the curve somewhere between the nodes and move this imaginary "inside" point. The curve's shape changes accordingly. We call this method "*non-node editing*". This means that you can move not just nodes, but every point of a glyph's contour. You can even switch off nodes and still be able to edit the contour as you wish.

To modify a curve or straight segment using the non-node editing method:

1. Move the mouse cursor onto the place on the segment that you want to move.

2. Hold down the left mouse button. You will see a small color point that will show you the temporary point that you are moving.

3. Drag the mouse and observe how the shape of the curve changes. After a few experiments (which can be undone) you will have enough experience to use this method of editing.

Several notes that you should remember:

1. In non-node editing, guiding objects are not sticky. So, temporary points do not snap to the grid, guidelines, hints or anything else.

2. If you choose a temporary point near one of the ends of a curve, you will move that end, not just change the curve's shape. This is a useful method to locate a curve's endpoints.

3. When you hold down the left mouse button to begin non-node editing, you will see that the final nodes of the curve as well as the control vectors appear, simplifying the editing of this segment.

4. If you want to highlight the line or curve but do not want to modify it, hold down the **CTRL** key while clicking on the line or curve.

If you drag a "point" on a curve, its control vectors may change direction:

To fix the direction of the control vectors, hold down the **ALT** key and double-click on the node. You will see the connection mark turn yellow. **ALT**+double-click on it again to remove the fixed state. Alternatively you can right-click on the node and use the **Fixed BCP Direction** option to control this feature.

Changing Connection Type

The type of connection between segments is very important in maintaining the smoothness of contours. Connections can be of two types: smooth and sharp.

If a connection is smooth the direction of the adjacent curve control vectors or of the curve control vector and line is collinear and the contour is smooth at the connection.

To change the type of connection

1. Make the node visible.

2.1 Double-click on the node with the left mouse button.

2.2 Right-click on the node and select the connection type in the context menu:

Deleting Nodes

To delete nodes using the Edit tool:

1.1 Begin moving the node by dragging it.

1.2 While holding down the left mouse button, click the right mouse button. The node will be removed.

2.1 Move the mouse cursor onto the node and click the right mouse button.

2.2 In the context menu choose the **Delete node** command.

⬊   Note: If you click the right mouse button while editing the curve using the non-node editing method or while you are moving the control points of a curve the curve will not be removed. Instead it will change to a straight line.

Deleting Lines and Curves

You can delete a whole line or curve with the Knife tool. Activate the Knife tool with the ✐ button on the Tools toolbar or press the **3** key on the keyboard. Hold down the **ALT** key and click on the line you want to delete. Note that with this method you will break the outline:
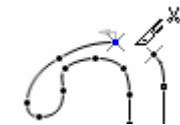
Another way to delete a line or a curve is to left-click on it with the Edit tool and then select the **Delete** command in the **Edit** menu. This method works differently because it does not break the outline.

Eraser Tool

The Eraser tool ✐ can be used to quickly remove nodes. Sometimes this is necessary, for example, with contours from an auto-tracing program. The eraser tool can work in two modes: like a standard eraser or as a rectangle eraser.

In the first mode, all nodes that are inside the eraser mouse cursor are deleted. In the second mode, you define a rectangle by clicking on and dragging (as when you select nodes with the Edit tool or change the zoom of a Glyph Window) and all the nodes inside the rectangle are removed.
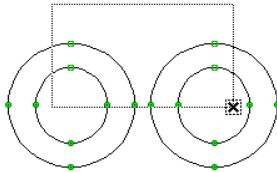
The first (eraser-like) mode is the default for the Erase tool. To switch to the rectangle mode, hold down the **CTRL** key.

Inserting Nodes

To insert a new node on a segment with the Edit tool:

1. Activate the Edit tool ↖ .

2. Position the mouse cursor on the segment where you want to insert the node.

3. Hold down the right mouse button. You will see a mark that shows you the current position of the mouse cursor. This mark will stick to all objects in the editing field, including nodes and glyph contours.

4. While still holding down the right mouse button, position the mark on the point where you want to insert the node and click the left mouse button. The new node will appear in that place. Release the right mouse button.

Using the Knife tool to insert nodes:

1. Activate the Knife tool ✎ .

2. Click on the point on the contour where you want to insert a node.

3. Hold down the left mouse button anywhere on the empty area of the editing field and drag the mouse to form a "knife line". After you release the left mouse button new nodes will be inserted at all points where this line crossed the outline. Hold down the **SHIFT** key to constrain the direction of the "knife line" to 45-degree increments.

   If the "knife" line will cross two lines you may find a part of the glyph to "cut off". Hold down the **ALT** key to limit Knife tool to insert new nodes only.

Using the Add Corner, Add Curve and Add Tangent tools:

1. Activate one of the tools.

2. Click on any outline point. The Corner tool will add a straight line, the Curve tool will add a smooth connection and curve and the Tangent tool will add a sharp connection and curve.

➲ Note: You can insert nodes on the closing straight segment that automatically connects the first and last nodes of a contour. If you insert nodes on the first half of a closing straight segment (closer to the ending node of a contour), then the new node will be added to the contour. If you insert the node on the last half of the closing straight segment, then it will be inserted before the startpoint and become a startpoint.

Using the Drawing Tool

The easiest way to create a new contour is to use the Drawing tool:

You can create a new contour or you can continue any existing contour. If you want to add new nodes to the existing contour, activate its first or next node:

The last node of the open contour is activated

1. To add a point, click the left mouse button.

2. If you want to create a line point, release the button. If you want to define a curve, drag the mouse to set the position of the curve control vector:

3. To adjust the position of the curve control vector without moving the control vector of the previous curve, hold **ALT** and left-drag:

    You can press and release the **ALT** key while you drag the mouse — when **ALT** is released you are defining the positions of the control vector that belongs to the previous curve *and* the control vector of the next curve. When **ALT** is pressed, you are not moving the previous curve's control vector.

4. When you are adding a new node, you can hold down the **CTRL** key to not move the curve control vector but move the node itself.

5. Finally, you can hold down the **SHIFT** key at any time to constrain the direction of the line (if you are holding down the **CTRL** key) or a curve control vector.

To close the contour, click on its starting node and drag the mouse to set the direction of the control vectors.

Adding Points to a Contour

In addition to the Drawing tool you can use three more tools to create a new contour or to add points to an existing contour. These tools are: Add Corner, Add Curve and Add Tangent.

To create a new contour:

1. Activate one of the tools.

2. Click the left mouse button anywhere in the empty area of the glyph window to create the first point of a new contour. Drag the mouse to put new node into correct position. Release the mouse button.

3. Click on again in the empty area to add a corner line, curve or smoothly connected curve (with Add Corner, Add Curve or Add Tangent tools respectively).

4. Continue the procedure until your newly defined contour is complete. To close contour, drag the last node onto the first node.

5. You can switch to the Drawing tool ⁺↕ at any time and use it to add new points to a contour you are creating.

❧ Note: A new node is added to the contour if the last node of the contour is highlighted. If it is not highlighted a new contour is started.

To highlight a node click on it. To deselect it, press the **Esc** key on the keyboard.

You can move outline nodes with the Add ... tools. Note that if you click on the contour (not on the node), a new node is inserted. The type of node depends on the tool you are using. To prevent adding a new node, hold down the **ALT KEY** when you click on the contour.

Converting Segments

Sometimes you may want to convert a curve to a straight segment or vice versa. **To convert a curve to a straight segment** "delete" (drag + right mouse button) one of the control points of the curve, or "delete" (drag + right mouse button) the curve while you are in the non-node editing mode.

To convert a straight segment (normal or closing) to a curve drag an inside point of the straight segment while holding down the ALT key.

To convert a curve to a 1/4 part of an ellipse (the curve's control vectors will be treated as an ellipse axis), press the ALT key and click on the curve.

You can also convert curves and straight segments with the context menu. Right-click on the end node of the segment and select the **Convert PS/TT** command in the context menu. With this command a line segment is converted to a Bézier curve, a Bézier curve to a TrueType curve and a TrueType curve to one or more Bézier curves.

The last way to convert segments is to use the selection menu:

1. SHIFT-click on any point on the curve or line segment.

2. Right-click on the highlighted (red) segment. A context menu appears.

3. Choose one of the commands in the **Convert** submenu: **To curves** (to convert to a Bézier curve) or **To lines** (to convert to straight line segments).

Breaking and Joining Contours

To break the contour with the Edit tool hold down C**TRL** and A**LT** and click on the node where you want to break the contour.

To break the contour with the Knife tool click on the node.

When a contour is broken its first and last nodes are highlighted by diagonal crosses:

You can use the Knife tool to "cut out" part of the contour:
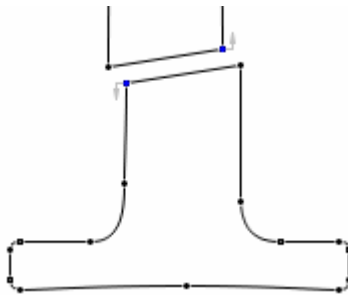
1. Activate the Knife tool.

2. Hold down the left mouse button and drag the cursor to define the "cutting line".

3. Release the left mouse button. Note that you can only cut part of a single contour, like in the sample image above.

To join two contours you need to move the starting or ending node of one contour to the starting or ending node of another contour.

Hold down the A**LT** key to prevent the contours from joining.

Node Commands

If you right-click on a node you will see a context menu with many useful commands:
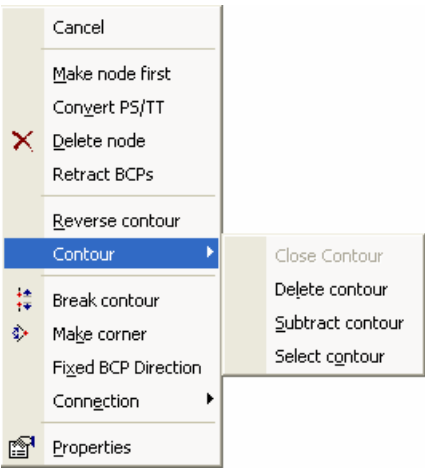
Below is a description of all the commands in this menu:

Make node first	Starts the current contour from the selected node (i.e. makes it the startpoint). This command is useful when you need to join contours since you can only connect starting and finishing nodes
Convert PS/TT	Cycles the node type from line to Bézier curve to TrueType curve
Delete node	Removes the node
Retract BCPs	Removes the control vectors of the node, making it sharp
Break contour	Breaks the contour at the selected node
Make corner	Makes a 90 degree corner (this operation is not always available)
Fixed BCP Direction	Makes the connection fixed. You can use it instead of **ALT**+double-clicking on the node
Contour	Set of commands related to the contour to which the selected node belongs (described below)
Connection	Popup menu with connection settings. You can use it instead of double-clicking on node
Properties	Opens the Node properties panel.

Contour commands:

Reverse contour	Reverses the contour direction
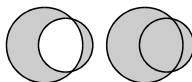
Close contour	Makes open contour closed
Delete contour	Removes the contour
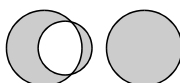
Subtract contour	"Subtracts contour" from the outline
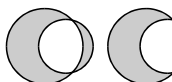
Select contour	Reverses the selection state of the contour

Node Properties

CTRL-click on the node and choose the **Properties** command in the context menu. You will see the Node properties panel:
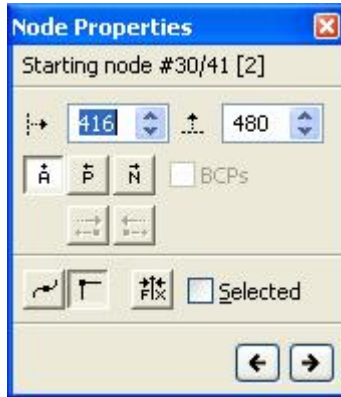
In this property panel you can control the position of the node, the alignment type, the selection status of the node and the position of the control points of the curves.

The figures in the first line are the index of the segment, the node index and the contour's number.

To change the position of a node:

1. Select the origin point you want to use and set the coordinates of the node. By default the origin is the glyph's origin point $\overset{\textstyle\cdot}{A}$. With the radio buttons you can select the previous $\overset{\textstyle\cdot}{P}$ or next $\overset{\textstyle\cdot}{N}$ node as the origin point.

2. Modify the coordinates of the node in the edit boxes. You can use the spin buttons to increase or decrease the coordinates. The new coordinates will be applied to the node when you press the **ENTER** key on the keyboard or move the focus from one edit control to another or when you close the property panel by clicking on a free space in the edit field.

To change the selection state of a node: modify the state in the **Selected** check box.

To change the connection mode for a node use these check boxes:

To edit the position of the curve's control vectors: switch on the **Control vectors** check box (it will be gray if you are editing a node between two straight segments) and modify the relative position of the previous or next control point that belongs to that node.

Use the buttons with arrows to edit the previous or next node.

➲ Tip: when you are editing node positions in the Properties panel, press the **ENTER** key to accept changes and move the focus to the editing field of the Glyph Window. There you can use the keyboard to move the selected node and the **PAGE DOWN**/**PAGE UP** keys to select another node for modification. You will see the node properties change in the Properties panel as you move the node by keyboard or mouse. Press **ALT**+**ENTER** to put the focus on the Properties panel to set the node position more precisely.

VectorPaint Mode

VectorPaint is Fontlab's unique set of tools that allow you to paint vector contours with tools that look and feel like bitmap tools. You can choose brushes, pens, freeform selections and even enter text. The idea of VectorPaint is that all the tools produce contours that combine with the existing glyph contours using our unique contour-processing technology.

When you click on one of the tools on the Tools or Paint toolbar you enter the VectorPaint mode. To open the Paint toolbar select it the **View > Toolbars** menu:
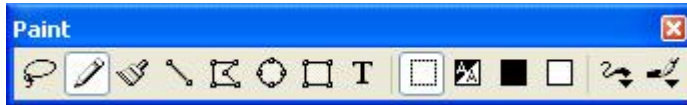
The keyboard shortcut for the mode is **ALT+3**.

The type of interaction between existing and new contours depends on the selected color mode. This process is very fast and is completely transparent to you, so if you switch on the preview mode (where the glyph appears filled), the illusion of bitmap-like editing of a contour-based glyph image is very realistic.

All the paint tools can work in 4 different color modes:

☐	**Transparent**	Newly created vector objects that are generated by the application of VectorPaint tools do not interact with the existing glyph's contour and appear selected for easy editing
▨	**Automatic**	The color of the brush depends on the point where you begin drawing. If you begin in a white area, a white brush will be selected, if in black, a black brush will be selected. Use this color mode to easily extend white or black areas of the glyph
■	**Black**	Generated contours are added to existing contours, expanding the black area of the glyph. It looks like a black brush applied to a black image
☐	**White**	New contours are subtracted from existing contours, simulating a white brush.

Here is an example of a brush stroke applied with Transparent, Black and White "colors":

Note: VectorPaint tools have an option to automatically activate the Free Transform operation when any of the painting operations is completed. This option allows you to instantly move, scale, rotate or slant the newly created shape.

Here is a list of all available VectorPaint tools with a short description of each:

	Freehand Select	Used to select non-rectangular areas of a glyph. It selects not the nodes, like the Edit tool, but actually cuts lines and curves and selects black areas that can be moved or otherwise transformed
	Pen	Used to create new contours or modify existing ones. It is not really a "paint" tool, because it deals with contours, but it is a very natural and flexible tool used to adjust the result of the application of VectorPaint tools
	Brush	Exactly that — a brush. It can be round or calligraphic. A calligraphic brush can be of any size and slant angle
	Line	Used to draw straight lines with a selected brush
	Polygon	Has two modes: point-by-point polygon drawing with easy combination of straight segments and curves, or point-by-point definition of a polygon that will be drawn by using the selected brush
	Ellipse	Used to draw ellipses or circles
	Rectangle	Used to draw rectangles or squares
T	Text	Used to enter text (vector based) using any TrueType font installed in the system.

Freehand Select Tool

This tool works like a precision knife. You can cut part of a contour, and it will be automatically selected so you can transform it, delete it or copy it.

To select part of a glyph with the freehand select tool:

1. Select the freehand select tool ( ) in the Paint toolbar.

2. Position the cursor on the point where you want to start the selection and hold down the left mouse button.

3. Drag the mouse while holding down the left mouse button to extend the selection polygon in freehand mode, or release and click the left mouse button to extend the selection polygon by straight segments.

4. Click the right mouse button to finish the selection.

When you finish the selection, you will see that the selection polygon was applied like a knife and you have a new contour (or several contours) that is separated from the glyph. The new contours are selected so you can use the Edit tool to move them or transform the selection. Of course, you can use any **Edit** menu command with this selection.

Pen (Contour) Tool

With the Contour tool ✏ you can create new contours or modify existing contours in a more artistic manner than with the Edit tool. When you use the Contour tool, you can draw new contours as you would do on paper. TypeTool will trace your drawing and replace it with a series of curves and lines.

How to create a new contour

If you begin a contour in a free area (where the cursor has its ordinary shape), you will define a new contour. If you want to begin a new contour but its startpoint is on an existing contour press the **CTRL** key to force TypeTool to create a new contour.

How to modify an existing contour

When you move the cursor of the Contour tool onto an existing contour or node, it changes. If you begin drawing (without holding down the **CTRL** key) the new contour will be inserted into the existing one. If the finishing point of your drawing is on an existing contour also, and the starting and finishing points are on the same contour, then the new drawing will replace the part of the existing contour that lies between the starting and finishing points.

How to draw a single curve

Hold the **SHIFT** key down when you release the left mouse button after drawing a new line. Your drawing will be approximated by a single curve. This is a good way to draw a new contour step-by-step.

Brush Tool

The Brush tool ✎ works like the usual bitmap brush that you find in any bitmap-editing program. You begin a brush stroke by holding down the left mouse button. Draw the stroke by dragging the mouse and finish drawing it by releasing the left mouse button.

To change the color of the brush, use the color selection buttons on the Paint toolbar:

☐	for the "empty" color
▨	for the "auto" color
■	for the "black" color
☐	for the "white" color

Other brush options are accessible in the **VectorPaint options** menu:

●	Round Brush
╱	Flat Brush
·	20 Units
●	40 Units
●	80 Units
●	120 Units
▤	Options...
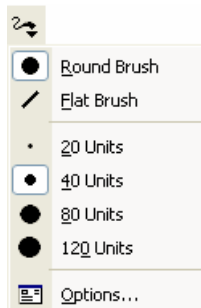
You can paint with round or calligraphic brushes of different widths:

╱	Calligraphic-style brush
●	Round brush
·	20 unit wide line
●	40 unit wide line
●	80 unit wide line
●	120 unit wide line

You can also specify a brush stroke shape. Press the ⬟ button and select a shape in the context menu:
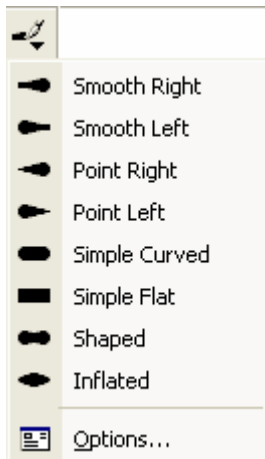
This is an example of different brush strokes:

VectorPaint Options

You can also change the brush properties in the Paint Options dialog box. To open the Paint Options dialog box, choose the **Options** command in the **Brush options** menu (²⁺) or **Brush style** menu (⁻²). You will see following dialog box:
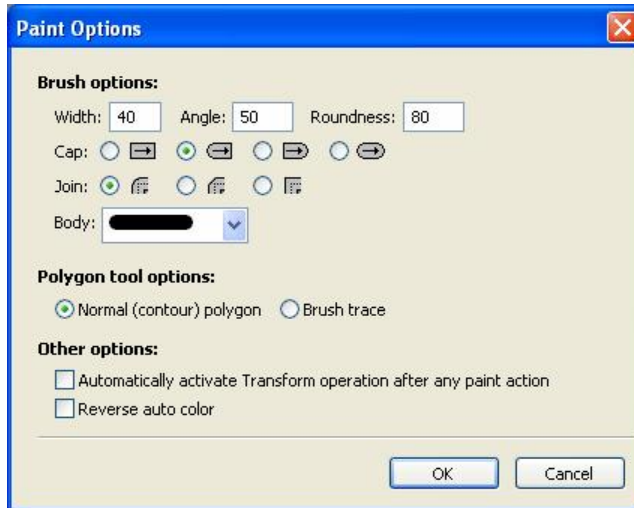
In the Paint Options dialog box you can enter the width of the brush and change the slant angle of a calligraphic brush. Additionally, you can select how the brush strokes are started and finished. Choose the brush's starting and finishing shape by activating one of the radio buttons.

You can also select the style of the connection between two sequential segments of brush strokes. It can be sharp, smooth or flat. Select one in the dialog box. Icons near the radio buttons give an explanation of the styles of connections.

Last field in the Brush options area lets you to select the shape of the brush stroke.

Other options relate to the Polygon tool (described later) and the options for automatic activation of the Free Transform operation and reversing of the "auto" color.

To choose one of the Polygon tool modes use the options in the Polygon tool field. If you choose **Normal (contour) Polygon** the polygon tool will create a simple contour that can include straight segments and curves. If you choose the **Brush trace** option the current brush will be applied to the created polygon's contour. In the Brush trace mode you cannot draw curves while defining a new polygon.

If you mark the **Automatically Activate Transform Operation** check box and select the Transparent painting color the Free Transform operation will be activated and applied to the contour that you created after the completion of any paint operation.

The last option, **Reverse auto color,** changes the behavior of the "auto" color brush mode. If you begin drawing in a white area, a black brush will be selected, if in black, a white brush will be selected.

Line Tool

The Line tool ✎ allows you to apply brushes to straight-line segments.

To draw a line segment:

1. Position the mouse cursor on the beginning point and hold down the left mouse button.

2. Move the mouse to the end point and release the button. Hold down the SHIFT key to constrain the direction of the line to 15-degree increments.

Polygon Tool

The Polygon tool ⌐ can be used in two modes: as a tool to draw a polygon consisting of lines and curves or as a tool to draw an outline of a polygon with a selected brush. The second mode can be treated as a series of applications of the line tool. The mode of the polygon tool can be selected in the **Paint Options** dialog box.

To draw a polygon using the Polygon tool:

1. Select the Polygon tool in the Paint toolbar. Be sure that the Polygon tool is in the polygon mode.

2. Move the mouse cursor to the first point of the polygon and click the left mouse button.

3. Move the mouse cursor to the position of the next polygon point. To add a line segment, click the left mouse button. To add a curve segment, press the **TAB** key, click the left mouse button and drag the mouse to define the control vector of a curve. Hold down the **SHIFT** key to constrain the direction of the control vectors to 15-degree increments.

4. If the polygon in its present state is finished by a curve, the next segment that the polygon tool will try to add will be a curve. To switch between adding a straight segment or a curve, press the **TAB** key on the keyboard.

5. Repeat steps 3 and 4 for all points of the polygon.

6. Click the right mouse button to finish creating the polygon.

The **Brush Trace** mode of the polygon tool works just like the normal (contour) mode.

Ellipse and Rectangle Tools

The Ellipse ⬡ and Rectangle ⬜ tools are very similar. The only difference is in the result.

To draw an ellipse or rectangle:

1. Select the tool that you want to use.

2. Position the mouse cursor on the spot where you want to place one of the rectangle corners (or on one of the corners of the rectangle that surrounds the ellipse). If you hold down the **ALT** key the mouse cursor will become the center of a rectangle or ellipse.

3. Hold down the left mouse button and drag the mouse to define the rectangle (or ellipse).

4. Hold down the **SHIFT** key to draw a square or a circle.

5. Release the button to finish creating the rectangle (ellipse).

Text Tool

With the Text tool you can add text to a glyph. Carefully select the color mode when planning to use the text tool. It is usually best to use the "Empty" color because in that mode the text stroke will not interact with the existing contour and you will be able to adjust its position using the Edit tool or the Free Transform operation.

Select the **Automatic Activation of Transform Operation** option in the Paint Options dialog box. With this option on the Free Transform operation will be activated immediately after entering the text string, allowing you to modify its size or position.

To enter a string of text:

1. Select the Text tool T in the Paint toolbar.

2. Position the mouse cursor (with the crosshair and the "suggested rectangle" of the future string) on the place in the editing field where you want to add the string.

3. Click the left mouse button.

4. In the dialog box, enter the character string. Use the **Font** button to select the font that will be used.
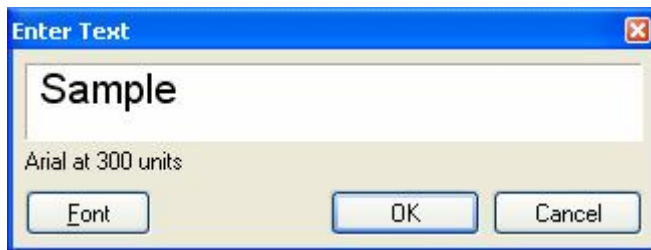
Below the sample string you will see the name of the current font and the size of the text string. The size is presented in font units. You can change the string size in the Font dialog box. The size of the placed text will be 10 times the selected point size. For instance, if you select a 24 pt. font you will get a string that will be 240 units in height.

5. Click on **OK** to enter the string or **Cancel** to abort this operation.

Selections

Many operations can be applied not only to single nodes or segments but also to several nodes together. For example, you may want to move many nodes or delete part of a contour. First, select the nodes that you want to process.

To select nodes with the selection rectangle:

1. Make sure that the Edit tool ![edit tool icon] is active.

2. Hold down the left mouse button anywhere in the empty area and drag the mouse to surround the nodes with a rectangle. Hold down the **SHIFT** key to reverse the selection state of the nodes.

To select or deselect individual nodes, **SHIFT**-click on them.

To select the contour segment (line or curve), **SHIFT**-click on it.

To select the whole contour double click on the contour (not the node) with the left mouse button or press the **CTRL** key and click anywhere on the empty area close to the contour. Hold down the **SHIFT** key to reverse the selection state of the contour's nodes.

To select all the contours in a glyph use the **Edit > Select all** command.

To deselect all nodes click the left mouse button somewhere in the free space of the editing field or use the **Edit > Deselect** command.

To reverse the selection state of all nodes in the glyph use the **Edit > Invert Selection** command.

199

Using the Magic Wand Tool

With the Magic Wand you can easily and precisely select contours. It is especially useful when you are working with glyphs that have many contours, such as Far-Eastern ideographs:
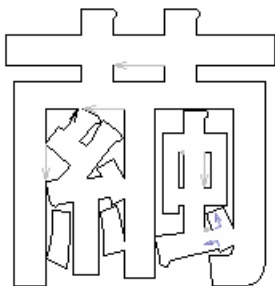
To select the contour with the Magic Wand tool, activate the tool (press the **4** key on the keyboard) and click anywhere near the contour. You do not need to be precise — TypeTool will automatically locate the closest contour.

To reverse the selection state of the contour, hold down the **SHIFT** key and click anywhere near it:
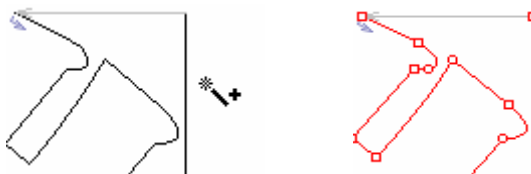
You can also select a contour and all contours that are inside it — hold down the **ALT** key when you using the Magic Wand tool:
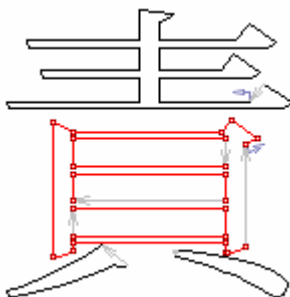
Moving the Selection

You can move the selected part of the contour by mouse — drag any selected part of the contour or use the arrow keys. If you press the ARROW key then the selection will move in that direction by one font unit. Hold down **SHIFT** or **CTRL** while pressing the arrow keys to accelerate the movement of the selection.

Selection Commands

When a part of the glyph is selected, right-click on it to get access to the context menu:
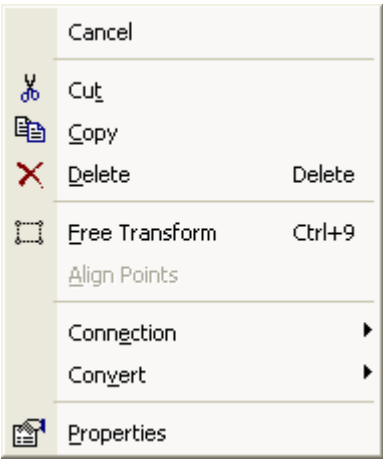
Some commands are duplicates of the **Edit** menu commands, but there are some unique commands:

Cut	Copies the selection to the Clipboard and removes it
Copy	Copies the selection to the Clipboard and leaves original untouched
Delete	Removes the selection
Free Transform	Activates the Free Transform operation
Align Points	Aligns the selected nodes vertically or horizontally
Properties	Opens the Selection properties panel.

There are some differences between the **Cut** and **Delete** commands. First of all **Delete** does not put anything on the Clipboard. But the main difference is that the **Delete** command removes nodes with their adjacent curves:

Selection   *Result of Cut*   *Result of Delete*
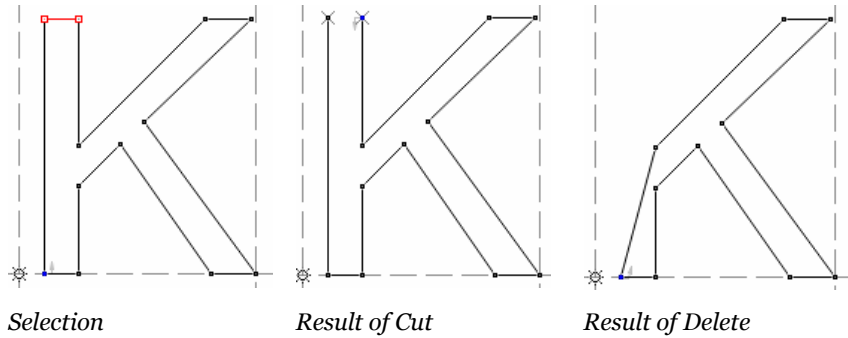
Selection Properties Panel

To make the Selection Properties panel visible, choose the **Properties** command in the selection context menu or use the **ALT-RETURN** keyboard shortcut.

The Selection Properties panel is very simple:

It contains the following information about the selection: the number of selected nodes, the number of selected contours, and the selection bounding box's bottom-left and top-right corner coordinates.

You can click on the **Deselect** button to discard the selection and get the Glyph Properties panel instead.

Copying the Selection

Sometimes you need to copy glyphs or parts of glyphs to another place in the font or even into a different font. With TypeTool you can put any part of a glyph or an entire glyph (with hints, guides, etc.) into the Windows Clipboard and paste it into a different place.

To copy parts of the glyph's outline use the commands from the **Edit** menu or the buttons on the Standard toolbar:

Cut	✂	To copy a selected part of the glyph onto the Clipboard and delete it from the glyph
Copy	📄	To copy a selected part of the glyph onto the Clipboard
Paste	📋	To add a contour part copied to the Clipboard into the current glyph as a new contour

Insert	To replace the current selection with the Clipboard contents
Delete	To remove the selected part of a glyph's contour
Duplicate	To insert a copy of the selection into the current glyph as a new contour
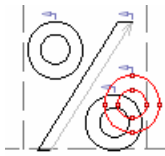
When you use the **Paste** command, the selection is pasted without offset from the original location. When you use the **Duplicate** command, the selection is pasted with offset [100,100].

Because the Clipboard is used as a buffer for copying contours you can paste glyphs and their parts not only to the current font but also to any glyph of any font of any application that is compatible with TypeTool (ScanFont 3.0, for example).

Transforming the Selection

Sometimes you need to scale, rotate or slant a whole or part of a glyph outline. In TypeTool you can do this using several methods:

1. Using the Transformation panel

2. Using the Transform tools

3. Using the Free Transform operation

↘ Note: To avoid naming confusions, former **Tools > Transform** feature has been renamed to **Action**. Starting from TypeTool 3.0, the user interface uses the terms "Transform" and "Transformation" to refer to geometric transformations such as rotation or scaling that can be applied to glyph outlines. Actions refer to operations that may affect outlines but also other font elements such as metrics or hints.

Using the Transformation Panel

The Transformation panel allows you to apply several simple transformations to the selected area or to the whole glyph. To open the Transformation panel you can select a **Transformation Panel** command in the **Window** menu:
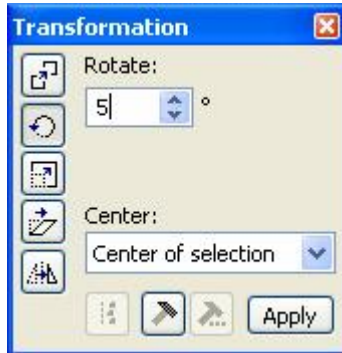
To transform the glyph or the selected area:

Select the type of the transformation by clicking on one of the buttons in the left and the transformation options in the right area:

	Shift	Shift (x, y): [0] [0] u
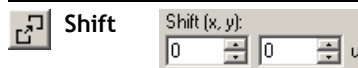
Enter a distance to move the selection in font units

	Rotate	Rotate: [0.0] ° Center: [Zero point]
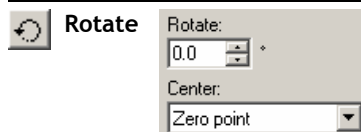
Enter the rotation angle (degrees, counterclockwise) and select a center of rotation:

Zero point
Center of selection
Bottom left corner
Reference point
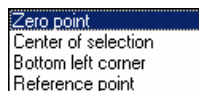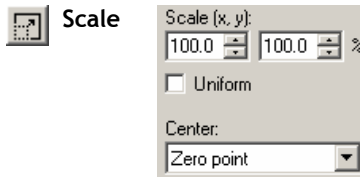
⬚	**Scale**	Scale (x, y): [100.0 ⬍] [100.0 ⬍] % ☐ Uniform Center: [Zero point ▼]
		Enter the scaling factor and select a center point of transformation. Use the Uniform option to scale proportionally
⬚	**Slant**	Slant: [0.0 ⬍] ° Center: [Zero point ▼]
		Enter the slant angle (degrees, positive value slants to the right) and select a center point
⬚	**Mirror**	Mirror axis: [0.0 ⬍] ° ⬚ ⬚ Center: [Center of selection ▼]
		Enter the direction of the mirror axis and select the center point of the transformation. Use the buttons to mirror horizontally or vertically quickly.
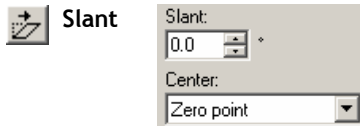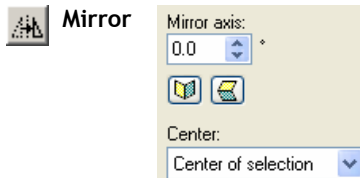
Click on the **Apply** button or press the **ENTER** key to apply the transformation to the selected area.

Pressing the ⬚ button will align all selected nodes horizontally or vertically.

If you press the ⬚ button the Actions dialog box will open (see the "*Actions* (on page 301)" chapter below). Pressing this button is the same as cho0sing the **Tools > Action** command. If you press the ⬚ button the previous transformation action will be repeated. It is the same as choosing the **Tools > Repeat Action** command.

207

Using Transform Tools

In the Edit mode you have access to three transform tools (in the Tools toolbar):

⟲	**Rotate**	Rotates the contour
⊡	**Scale**	Scales the contour
⟁	**Slant**	Slants the contour vertically or horizontally.

To transform the outline:

1. Select part of the outline you want to transform or undo all selections to transform the entire glyph outline.

2. Activate one of the transform tools.

3. Position the mouse cursor at the center of transformation, hold down the left mouse button and drag the mouse to make the transformation. Remember that you can hold down the '`' key (the key under **Esc**) at any time to get a high-quality preview of the transformed glyph.

4. Use the **Shift** and **Ctrl** keys to constrain the transformation.

5. Release the left mouse button to complete the transformation of the outline.

Using the Free Transform Operation

To activate the Free Transform operation select the **Free Transform** command from the **Contour > Transform** menu or click on the ![button] button in the Tools toolbar.

Or you can double-click on any selected (red) segment to activate the Free Transform operation.

When this operation is activated, you will see a transformation rectangle surrounding the selected area. If nothing is selected, the entire glyph will be subject to transformation.
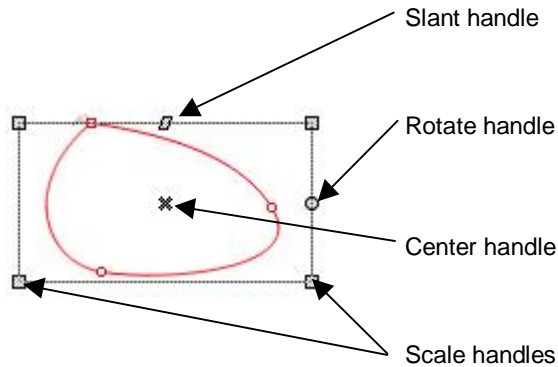
So, what do all these handles mean, and how can they be used?

To move a selection:

1. Position the mouse cursor somewhere inside the transformation rectangle but not on the center handle.

2. Hold down the left mouse button and drag the rectangle to its new place.

3. Release the left mouse button. The selection will be moved.

To scale or skew a selection:

1. Position the mouse cursor on one of the scale handles □.

2. Hold down the left mouse button and drag the mouse. You will see that the transformation rectangle is scaled. Hold down the **SHIFT** key on the keyboard to constrain the scale proportionally.

3. Release the button when you are done. The selection will be modified.

To rotate a selection:

1. Move the mouse cursor onto the rotation handle ○.

2. Hold down the left mouse button and drag the mouse. The transformation rectangle will rotate around its center. Hold down the **SHIFT** key to constrain the rotation angle to 15-degree increments. You can also use the rotation handle for slant — press the **CTRL** key to alternate between rotate or slant.

3. Release the button to accept the rotation.

To move the center of rotation, drag the center handle ✳ by the mouse to its new position.

To slant a selection:

1. Move the mouse cursor onto the slant handle ▱.

2. Hold down the left mouse button and drag the mouse. The transformation rectangle will be slanted. Hold down the **SHIFT** key to constrain the slant angle to 15-degree increments. You can also use the slant handle for rotation — press the **CTRL** key to alternate between rotate or slant.

3. To slant in vertical direction, hold down the **CTRL** key and drag the rotate handle:
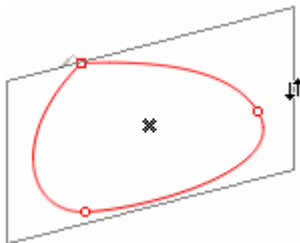
4. Release the button to accept the slanting.

Double-click on in the editing field or press **ENTER** to accept the completed transformation or press the **ESC** key to reject it.

You can use the arrow keys while the Free Transform operation is active to move the selection by one font unit in the direction of the arrow key you pressed. **SHIFT+ARROW** keys move the selection by 10 font units at each keystroke. **CTRL+ARROW** keys move the selection by 100 font units at each keystroke.

Building an Outline from Blocks

Now you know how to select parts of an outline and copy it, so let's do a few experiments to show how to use this knowledge.

Suppose that we have an 'I' glyph and we want to create an 'H' glyph.

1. Open the 'I' in the Glyph Window (double-click on the 'I' cell in the Font Window or browse the font with the arrow buttons).

2. Cut the glyph in the middle. Activate the Knife tool, click the left mouse button on the left of the glyph, hold down the **ALT KEY** on the keyboard and drag the mouse cursor to the right to define a cutting line. Release the left mouse button:
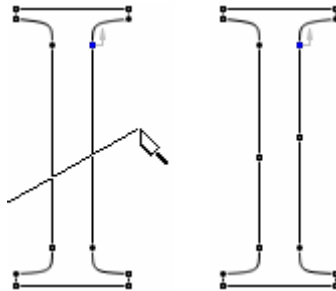
3. Click on each inserted node to break the contour:
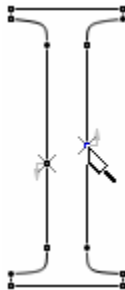
4. Select the bottom half of the 'I'. Temporarily activate the Edit tool (tap on the **CTRL** key on the keyboard) and **double**-click on the bottom of the glyph:
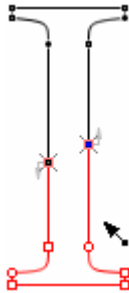
5. Copy it to the Clipboard with the **Edit > Copy** command.

6. Go to the 'H' glyph.

7. Use the **Edit>Paste** command to place a copy of the 'I' bottom:
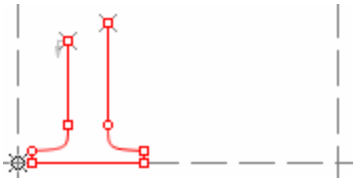
8. Use the **Edit>Paste** command to make a second copy. Shift-drag the second copy horizontally:
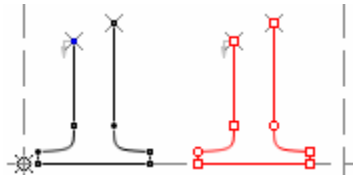
9. Copy and paste it again and shift-drag it to a place above the second segment:
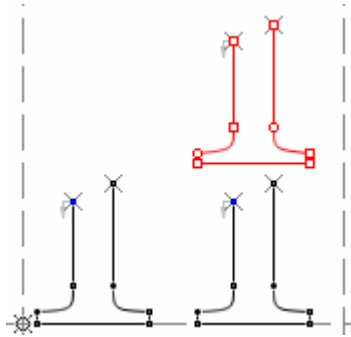
Use the contour snap function to position the segment. Activate the contour snap with the **View > Snap to Layers > Outline**. We also recommend that you activate the feature that will snap a point which you are moving to all outline nodes, by X and Y direction independently. Use the Tools > Options > Glyph Window dialog box:

☑ Align to all contour points if snap to contour is on

10. Use the **Contour > Transform > Flip Vertical** command to flip the selected segment:
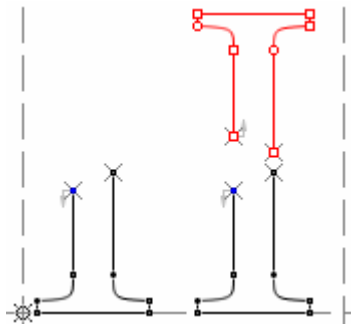
11. Duplicate the top segment and locate the copy above the bottom-right segment.
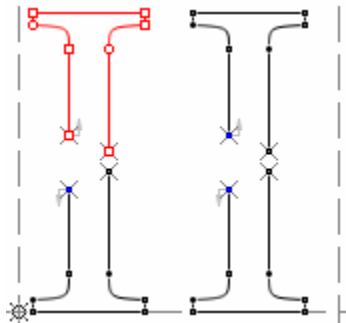
12. Left-click on an empty area of the Glyph window and then click on the bottom-left node of the top-left segment to highlight it::
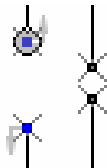
13. Activate the Add Corner tool ⊤ click on somewhere and drag the line to connect the left line of the top-left and bottom-left segments:
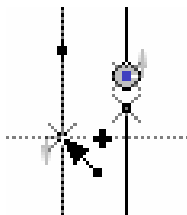
14. Then click on any other starting or ending node of the contour segments and use the Add Corner tool to connect them:
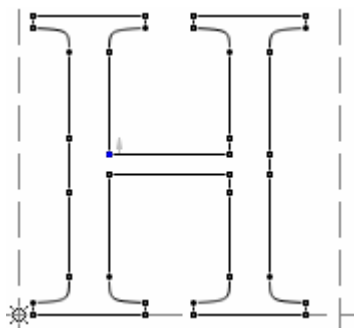
you will notice it takes more time to read the instructions than to actually perform the procedure.

Contour-related Commands

You can find the commands described below in the **Contour** menu:

|		
🗺	**Flip Horizontal**	Makes a mirror transformation in the horizontal direction. This operation is applicable to a selection or to a whole outline if nothing is selected
▨	**Flip Vertical**	Makes a mirror transformation in the vertical direction. This operation is applicable to a selection or to a whole outline if nothing is selected
⊕	**Merge Contours**	Combines all overlapping parts of the outline. This operation and the two following operations are applied to all contours that have at least one node selected. If nothing is selected, they will be applied to the whole glyph
⊟	**Get Intersection**	Leaves only areas that are covered by at least two contours
⟡	**Set PS Direction**	Sets the direction of all curves to PostScript curves (black on the left)
⟡	**Set TT Direction**	Sets the direction of all curves to TrueType curves (black on the right)
⟍	**Reverse All Paths**	Reverses the direction of all contours of the glyph
⟶𝖳𝗍	**Curves to TrueType**	Converts all Type 1 (3rd-order) curves to TrueType (2nd-order) curves
⟶𝖳𝟣	**Curves to PostScript**	Converts all TrueType (2nd-order) curves to Type 1 (3rd-order) curves
♂	**Correct Connections**	Analyses an outline and fixes the types of connections between outline segments (lines and curves)
杝	**Join Broken Contours**	Automatically joins all "broken" contours in a glyph. This command may not work if the nodes that should be connected are not close enough to each other
	Close Open Contours	Closes all open contours in a glyph creating a straight segment between first and last nodes
⌿	**Nodes at extremes**	Automatically inserts nodes at the extreme points of curves. We highly recommend that all curves have nodes at their extreme points
⊁	**Align to Guides**	Move all nodes on to guidelines, hints or grid if they are sufficiently close to them. This command will "snap" nodes only to the guiding layers that are currently visible.

Most of these commands can be applied to many glyphs at a time if you select a number of glyphs in the Font Window. Commands marked with the » sign in the **Contour** menu are tools and were not included in the table above.

Merging and Intersecting Contours

With the **Merge Contours** and **Get Intersection** commands, which are available in the **Contour > Transform** menu, you can perform very interesting operations on contours.

All two operations are applied to contours that have at least one node selected or to the whole glyph outline if nothing is selected.

The **Merge Contours** command combines contours, removing all outline overlapping and keeping the filled result unchanged:
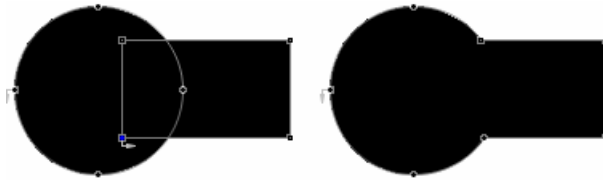
The **Merge Contours** command is the outline equivalent of the Boolean "OR" operation.

The **Get Intersection** command will keep only the area of intersection, removing all other parts:
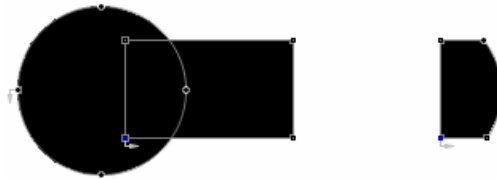
This command is the outline equivalent of the Boolean "AND" operation.

➲ Note: You can select multiple glyphs, or all glyphs, in the Font Window and apply the commands to many glyphs at a time.

Converting Contours

Several commands in the **Contour** menu are used to change contours' attributes.

Sometimes, you may find that e.g. the interior counter of "o" is black instead of white/transparent, or two overlapping contours create a white intersection. This means that one of the contours in your glyphs has a wrong direction.

TypeTool can detect such problems and automatically correct the contour direction for PostScript (Type 1/Bézier) or TrueType outlines. For that, use the **Set PS Direction** (for Type 1 or OpenType PS contours) or **Set TT Direction** (for TrueType / OpenType TT contours) commands in the **Contour > Paths** menu. These commands will reverse some of the contours so that all contours have the contour direction prescribed by the font format.

To reverse *all* contours in the glyph, i.e. change the direction of every contour in the glyph to the opposite, use the **Reverse All Paths** command in the **Contour > Paths** menu.

Use the **Curves to PostScript** and **Curves to TrueType** commands in the **Contour > Convert** menu to convert glyph's outline to 3rd-order curves or 2nd-order curves respectively. These commands only convert curves but do not change their direction so you may need to use the **Set PS Direction** and **Set TT Direction** commands later.

➘ Note: You can select multiple glyphs, or all glyphs, in the Font Window and apply the commands to many glyphs at a time.

Grid Layer

This layer is very simple: if the Grid is on, you will see a grid of vertical and horizontal lines in the edit Window. If **View > Layers > Snap to Layers > Grid** is enabled (which it is by default) any node that you move will snap to the gridlines.
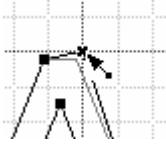

You can adjust the grid frequency on the **Glyph Window > Dimensions** page of the Tools > Options dialog box:

Guidelines Layer

Guidelines are straight lines that are used to guide the drawing of specific elements of a glyph. Guidelines can be vertical, horizontal or slanted.

Guidelines can be slanted at any angle from -45 to +45 from the vertical or horizontal direction. Slanted guidelines can help to mark *italic* glyphs, or specific slanted elements in normal glyphs, like the inner bar in the letter 'N'.
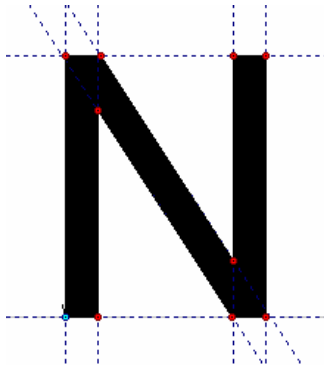
You can see little numbers giving the position and slanting angle of each guideline near the edges of the editing field of the Glyph window where the guidelines cross the rulers.

There are *local* and *global* guidelines. Local guidelines appear only in the glyph where they were set. Global guidelines appear in all glyphs of the font. Global guidelines are very useful to mark important levels in the font (by using horizontal global guidelines) or to set the base direction of an italic or oblique font (using slanted guidelines).

Editing Guidelines

Be sure that the Edit ↖ tool is active and the guidelines layer is visible —
use the **View > Show Layers > Guidelines** command to switch it on. Note
that the Guidelines layer will automatically switch on if you add a new
guideline.

To add a new guideline:

1. Position the mouse cursor on the horizontal ruler bar (for a horizontal
guideline) or on the vertical bar (for a vertical guideline).

2. Hold down the left mouse button. The bar will appear "pressed" and
the new guideline will appear. Hold down the SHIFT key to add the
global guideline.

3. While holding down the left mouse button, drag the guideline to the
desired place and release the button.

To move the guideline:

1. Move the mouse cursor onto the guideline that you want to move. Be
sure that no other objects (such as nodes or hints) are near the cursor.

2. Hold down the left mouse button and drag the guideline to the new
place.

While you are dragging the guideline and the mouse cursor is within the
snap-to distance the guideline will stick to the node. Nodes must be visible.

The guideline will snap to all nodes regardless of the mouse cursor position
if the option **View > Snap to > Outline** is on and the following check box
on the Glyph Window page of the Options dialog box is also switched on:

☑ Align to all contour points if snap to contour is on

To slant the guideline:

1. Move the cursor onto the guideline near one of the sides of the editing field of the Glyph window.

2. Hold down the left mouse button. The mouse cursor will change to a pair of curved arrows ⟲ that shows you the guideline slant direction.

3. Moving the mouse, slant the guideline to the angle that you want. Hold down the **Shift** key to constrain the slanting angle to 3-degree increments.

To remove the guideline:

1.1. Start moving or slanting the guideline.

1.2 While holding down the left mouse button, click the right mouse button.

2.1 Position the cursor on the guideline and click the right mouse button.

2.2 In the menu, select the **Delete** command.

You can use this option:

☑ Remove hints and guides by moving out of the window

located on the **Glyph Window** page of the Options dialog box, to remove any guideline or hint by simply dragging it from the editing field of the Glyph window.

To remove all local guidelines use the **Remove guidelines** command in the **Tools > Hints & Guides** menu. Options of this command include:

Both	to remove all guidelines
Vertical	to remove only vertical guidelines
Horizontal	to remove only horizontal guidelines.

The same command is available in the rulers context menu that appears if you right-click on the vertical or horizontal ruler.

Guidelines Popup Menu

More commands are available in the guideline's context menu.

The **Properties** command, as usual, will open the Property panel for the active guideline. The **Delete** command will remove the active guideline. The **Align** command is available only for slanted guidelines and will align them to the vertical or horizontal axis (i.e. remove their slant and make them vertical or horizontal guidelines).

Guidelines Properties Panel

To open the guideline properties panel, right-click on the guideline and choose the **Properties** command from the context menu:
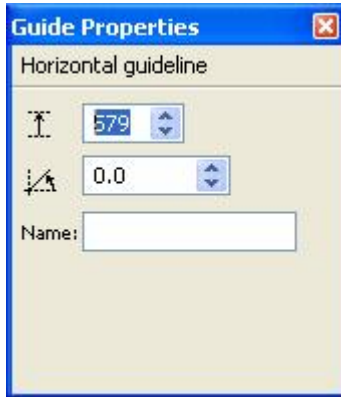
In this properties panel you can change the position and slant angle of a guideline. You also can name the guideline. This will help you to distinguish different guidelines.

To name the guideline, enter some text in the Name field and press **ENTER**. The guideline will get a label:
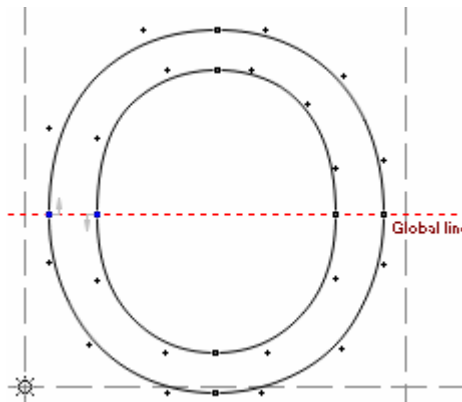
Guidelines' default colors are set on the **Options > Glyph Window > Colors** page.

Meter Mode

With the Meter tool you can measure any distance and angle in your glyph. It is very useful if you want to create very precise, extremely high quality glyphs.

To measure distances between two points:

1. Select the Meter tool ✎ in the Tools toolbar. The Meter tool panel appears:

This is a brief description of the fields on the Meter panel:

► x 2188 y -1509	Absolute position of the point (relative to the glyph zero point)
►₊ x 2188 y -1509	Reference distance (relative to the position of the reference point)
•► ⊢⊣ 545 ⊥ 260	Horizontal and vertical distance (from the beginning to the end of the metering line)
↙↗ ⟋ 603.8 ◺ 25.5	Geometric distance an angle of the metering line
:: ⁂ 23 ⁂ 4	The total quantity of nodes and of selected nodes in the glyph.

Note that you can open the panel at any time if you click on the I button in the bottom-left corner of the Glyph window. A second click on this button will close the panel.

2. Position the mouse cursor on your first point.

3. Hold down the left mouse button and drag the mouse to the second point. In the Meter panel you will see the vertical, horizontal and direct distance between two points and the angle of a vector that would theoretically connect these points. Hold down the **SHIFT** key while you drag the mouse to constrain the measurement to 15-degree increments.

You may dock the Meter panel to any edge of the Glyph Window.

227

While you are dragging the mouse you will see that the Meter tool arrow sticks to any object that it can find in the editing field.

To measure the distance from a contour:

1. Put the mouse cursor on the contour from which you want to measure.

2. Hold down the left mouse button and drag the mouse to what you want to measure to. Hold down the **SHIFT** key and the direction of the mouse's movement will be constrained to the normal direction of the contour startpoint.

3. When you are done, release the button.

Setting Guidelines

With the Meter tool you can not only measure angles and distances but also mark glyph elements with guidelines.

Hold down the right mouse button instead of the left mouse button and measure the distance. When you release the button a context menu appears.

Here is what you can do:

Add slanted guideline	A slanted guideline will be added along the meter tool's arrow. Note that the next guideline that you drag from the rulers will be parallel to this one

Background Layer

When other methods are not adequate you can use a *background bitmap* template. A bitmap template is a black-white bitmap image that appears on the screen underneath all the other layers. You can use it as a template for a glyph outline (it is especially useful when working with the VectorPaint tools).

To see the background layer switch it on in the **View > Show Layers** menu.

Create a background layer using any of the three following methods:

1. Open a bitmap image file (in BMP or TIFF format).

2. Paste an image from the Clipboard.

3. Rasterize the current outline to make an image in the bitmap background layer.

You can also copy the contents of the background to the clipboard to paste it into any Windows image-editing program; save it to the image file; and set its size and position on the screen.

To open a bitmap image, select the **Background** command from the **File > Import** menu:

Import	▶	Background...
Export	▶	Bitmap Font...
		EPS...
		Metrics...
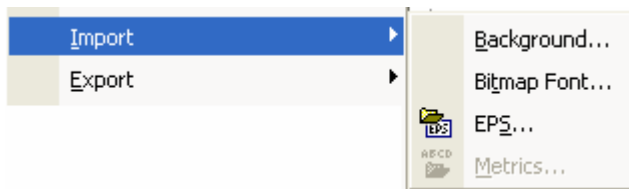
You will see the standard Windows Open File dialog box where you can select the bitmap file that you want to put into the background layer. Use the format selection combo box to choose the format of the bitmap file. TypeTool supports two formats: BMP (standard Windows bitmap format) and TIFF (standard image interchange format). Bitmap files that you import into TypeTool must be black and white (line-art) images. Neither color nor grayscale images can be imported into TypeTool. Your image editing application can usually make this change.

To copy a bitmap image from another Windows program into TypeTool, select the image in the program using its selection tools; copy the image onto the Clipboard (the image may be color, black-white or grayscale); switch to the TypeTool window; and select the **Paste** command from the **Edit** menu.

To rasterize a glyph's outline and make a background layer from it, select the **Create** command from the **Tools > Background** menu.

Below is a table containing all the commands from the **Background** menu related to background bitmap layer:

Create	Rasterizes the outline and makes a background layer
Copy	Copies the contents of the background layer to the Windows Clipboard. You can also use the Paste command from the Edit menu to paste the bitmap contents of the Clipboard to the background layer
Remove	Removes the contents of the background layer
Move and Scale	Activates the Bitmap Positioning operation described in the next section.

You can change the color which is used to render the bitmap background in the Glyph window on the Colors page of the Tools > Options dialog box described in the "*TypeTool Options* (on page 56)" section.

Background Positioning

This operation lets you set the size and position of the background layer:
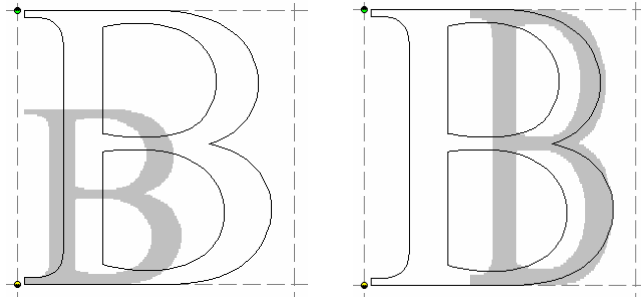
Different sizes and positions of the bitmap background layer

To set size and position of the background layer:

1. Activate the Bitmap Positioning operation. Select the **Move and Scale** command from the **Tools > Background** menu or press the ⊞ button on the Background toolbar or simply double click on the bitmap background while the Edit tool is active.

2. You will see a control box surrounding the bitmap.

3. Drag the handles in the corner of the control box to scale the background. Hold the **SHIFT** key to keep the proportions.

4. Position the mouse inside the control box, hold down the left mouse button and drag the mouse to position the background.

5. Press the **ARROW** or **SHIFT+ARROW** keys to move the background.

Press the **ENTER** key on the keyboard to finish positioning the background or the **ESC** key to cancel changes.

Outline Operations

In TypeTool, operations are temporary tools that let you modify your glyph. Operations are activated by pressing on their buttons in the Tools toolbar or by selecting a command in the **Contour** and **Tools** menus.

When an operation is activated one or more handles appear depending on the operation. After you make changes double-click to accept them (you can also press the **RETURN** or **ENTER** key on the keyboard) or press the **ESC** key to reject the changes.

When the operation is completed the tool that was selected before will be activated again. As with all permanent tools you can use the zoom selection tool, quick zoom keys and all the other viewing options of the Glyph Window while you are working with the operations tool.

Here is a list of all available operations:

⊞	**Free Transform (Contour > Transform > Free Transform)**	Scales, rotates or skews the selected portion of the outline or the whole glyph (*Free Transform* (on page 209))
⦂	**Position Background (Tools > Background > Move and Scale)**	Sets the size and position of the bitmap background layer (*Background Positioning* (on page 231)).

Metrics

The Metric data of a glyph includes information about the horizontal and vertical width. Glyphs have an *origin point*, a *baseline*, an *advance width* (or *character width*), *sidebearings*, and *left and right margins*:
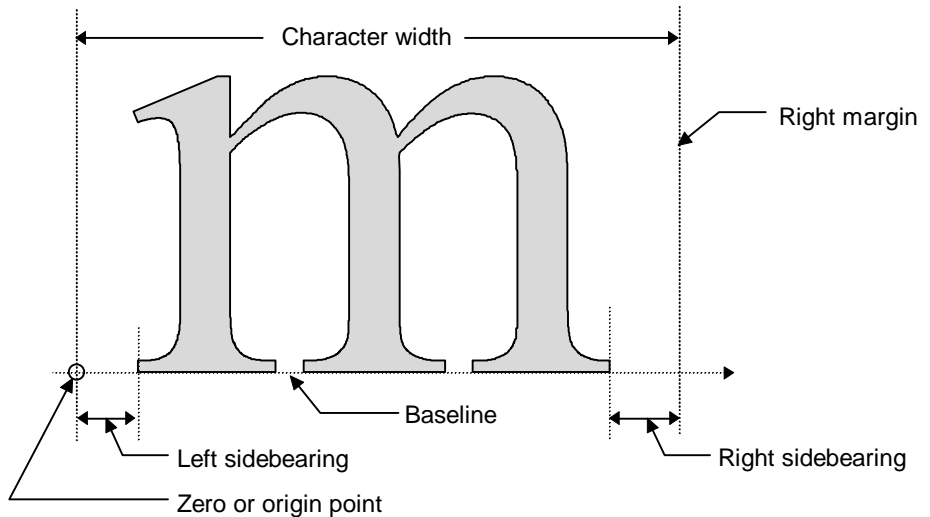
The baseline is used to align glyphs in a series. The left and right margins are used to define the positions of sequential glyphs in a series when the horizontal writing mode is selected. In the vertical writing mode the left and right margins are used to horizontally align glyphs and the top margin is used to vertically align glyphs.

In TypeTool, the position of the origin point is the position of the left margin in the horizontal direction and the position of the baseline in the vertical direction. However, you can modify the position of any of the four margins. If you move the baseline or left margin line you will shift the entire glyph.

Editing Metrics

TypeTool has a special window for editing glyph metrics, of course, but you can make small adjustments right in the Glyph Window, using the main edit tool.

Use the mouse and drag the left or right sidebearing or the baseline.

In TypeTool you can define vertical glyph metrics: the vertical advance "width" (called *vertical advance vector*) for Asian glyphs used to type in vertical direction from top to bottom. To define a vertical glyph advance vector, hold down the **SHIFT** key while moving the base line:
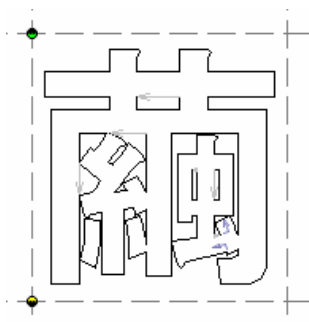
Baseline Properties Panel

With this property panel you can adjust the position of the glyph's baseline. To open it click the right mouse button on the baseline and select the ⊞ Properties command in the context menu or left-click on the baseline while holding down the **CTRL** key on the keyboard.
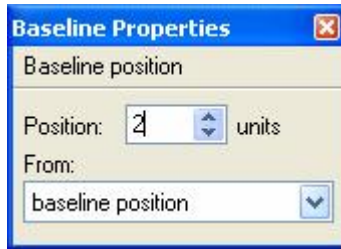
To change the position of the baseline:

1. Select the base level of the modification. It can be the old position (for relative offset) the top of the glyph, the bottom of the glyph, the top sidebearing, or the bottom sidebearing.

2. Change the position of the baseline relative to the base level.

3. Press the **ENTER** key or click anywhere in the editing field to apply the changes.

Metrics Properties Panel

To open the metrics property panel, position the mouse cursor on the left or right glyph margin, click the right mouse button and select the **Properties** command ⊞ Properties , or **CTRL**-click the left mouse button on one of the margins.
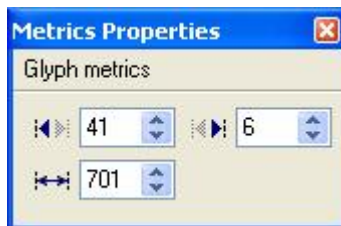
In this panel you can modify a glyph's sidebearings and/or advance width.

235

Mask Layer

When you need something more than guidelines or a grid to help with glyph editing you can use the *mask layer*. The mask layer is an outline that is created with the same segments as the glyph's outline. It appears in the Glyph Window as a dashed outline and the glyph's nodes "stick" to the mask. You can think of the mask as a "freeform" guideline.

The mask layer is very useful when you want to use glyphs of one font as a template for another font. For example, you can put the sans-serif version of the typeface into the mask layer while you are working on the serif version in the outline layer.

The mask layer can be filled by copying the selected part of the outline to the mask layer.

To copy the selected part of the outline to the Mask layer use the **Copy Outline to Mask** command in the **Tools > Mask** menu. If nothing is selected in the outline layer the entire glyph outline will be copied.

You can customize colors of the Mask layer background and outlines on the **Glyph Window > Colors** page of the Tools > Options dialog box described in the "*TypeTool Options* (on page 56)" section.

Editing Mask

To edit the mask layer with the usual editing tools you need to activate it with the **Edit Mask** command in the **View > Show Layers** menu.

When the Mask layer is selected for editing, the outline layer will be shown as a mask and may be filled, if Fill Outline (preview) mode is active, so you can use it as a reference. The editing field changes its color to remind you are in the Mask layer. Use any tool of the Edit mode to create, edit or remove the nodes and contours of the Mask layer outline. (Re)Activate the Outline layer when you are finished working on the mask.

You can switch to the Mask layer and back to the Outline layer simply by double-clicking on contours of the mask and outline. The editing field background color will change accordingly showing you whether you are in the mask-editing mode or not.

Mask Operations

All operations related to the Mask layer appear in the **Tools > Mask** menu:

Paste Mask to Outline

Adds the contents of the mask layer to the outline. The added part will be selected so you can start to work with it immediately.

Clear Mask

Clears the mask layer, removing all its contents.

Swap Outline with Mask

Exchanges the Outline layer and the Mask layer.

Vertical Metrics

Every font has several vertical font metrics for alignment of text:
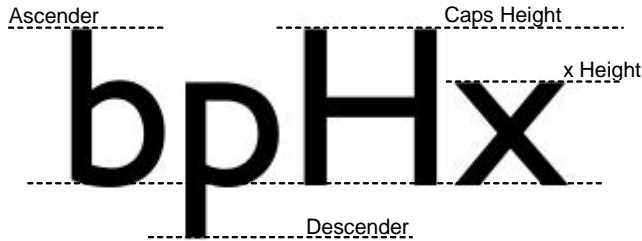
The **Ascender** line defines the position of the top of lowercase glyphs (usually the topmost point of the Latin 'b').

The **Descender** line defines the position of the bottom of the lowercase glyphs (usually the bottom point of 'p').

The **Caps height** defines the height of the uppercase glyphs (without overshoot). Usually it is the height of the 'H'.

The **x Height** is the height of most lowercase glyphs, like 'x' or 'v'.

In TypeTool you can modify the vertical metrics values in the Font Info dialog box, but you can also preview and change them visually in the Glyph Window.

Make sure that the Vertical metrics layer ⌗ is active and not locked.

In the Editing field vertical metrics appear as gray lines with a label at the left:

  Ascender

  Descender

  Caps height

  x Height

To change a metric, drag its line with the Edit tool or **Ctrl**-click on the metric line to open its properties panel and enter a numeric value.

In TypeTool you can also define vertical glyph metrics: the vertical advance "width" (called *vertical advance vector*) for Asian glyphs used to type in vertical direction from top to bottom. To define a vertical glyph advance vector, hold down the **Shift** key while moving the base line.

Hints Layer

Hints are used by the font rasterizer to improve a glyph's appearance on devices with low output resolution, like computer monitors or low-res printers.

There are two hinting methods applied to Type 1 fonts (hints for True Type fonts are always generated automatically): font-level hinting and glyph-level hinting. Font-level hinting is generated automatically in TypeTool, so you do not have to edit it manually. In the Glyph Window you can see Type 1 glyph-level hints.

Glyph-level hinting is applied to the glyphs' stems:

Vertical stems
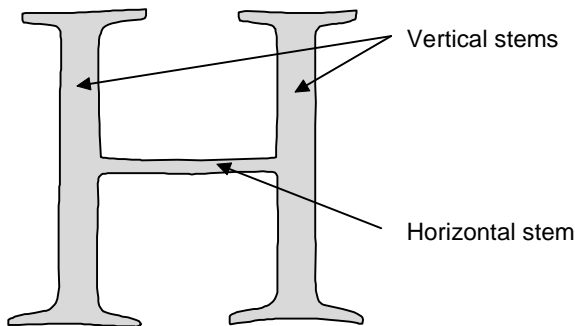
Horizontal stem

All important stems in a glyph should have stem hints, a pair of vertical or horizontal lines. The information in the hint includes not just the position of each of the two lines that "build" the hint, but also the position of one (major) line and the width of the hint.

You can declare stem hints in TypeTool by dragging them and modifying their width. Because hints in TypeTool are very "intelligent," they automatically snap to the contour, minimising your work. In most cases the autohinting algorithm that is included in TypeTool produces good results — usually not any worse than the results of manual hinting.

Editing Hints

Editing hints is very similar to editing guidelines. You can add new hints through the ruler bar of the Glyph Window; drag them with the mouse; and delete them by using the menu command or by clicking on both mouse buttons.

In contrast to guides, hints consist of two lines that can be moved together or separately. Hints cannot be slanted.

To add a new hint:

1. Position the mouse cursor on the horizontal ruler bar (for a horizontal hint) or on the vertical bar (for a vertical hint).

2. Hold down the CTRL key. Hold down the left mouse button. The bar will appear "pressed" and a new hint will appear. Release the CTRL key.

3. While holding down the left mouse button, drag the hint to the desired place and release the button.

To move a hint:

1. Move the mouse cursor onto one of the hint's lines.

2. Hold down the left mouse button and drag the hint to its new place. Both hint lines will move together.

To move the hint lines separately hold down the SHIFT key while dragging one of the hint's lines. Using this procedure you can change the width of the hint.

While you are dragging the hint and the mouse cursor is within the snap-to distance the hint line will stick to the node. Nodes must be visible.

The hint will snap to all nodes regardless of the mouse cursor position if the option **View > Snap to > Outline** is on and the following check box on the **Glyph Window** page of the Options dialog box is also switched on:

☑ Align to all contour points if snap to contour is on

↘ Note: While you are editing the hint, its parameters are shown on the status bar.

To remove a hint:

1.1 Start editing the hint.

1.2 While holding down the left mouse button, click the right mouse button.

2.1 Position the cursor on the hint and click the right mouse button.

2.2 Select the **Delete** command from the menu.

Hint Popup Menu

To open the hint context menu, right-click one of the hint lines.

The **Hint** context menu includes the following commands:

Reverse	Reverses the direction of the hint
Delete	Removes the hint
Properties	Opens the hint property panel.

Hint Commands

The **Tools > Hints & Guides** menu contains several commands related to hints:

Remove Hints	Removes vertical or horizontal or all hints and links. This command is duplicated in the rulers context menu
Autohinting	Automatically generates hints for the current glyph.

Hint Properties Panel

To open the hint properties panel, **CTRL**-click one of the hint lines or right-click on and choose the 🖳 Properties command in the context menu :
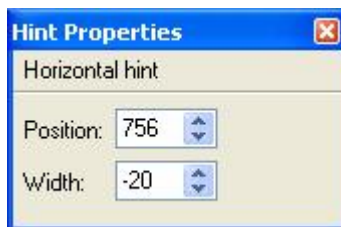
In the hint property panel, you can modify the position of a hint in the upper edit box and modify the width of the hint in the lower box. Press the **ENTER** key or click on the mouse outside of the properties panel to apply the changes.

Working with Composite Glyphs

Composite glyphs are glyphs made up of two or more components, like a letter plus an accent. One or more of the components are referenced. I.e. their contours are not actually present in the composite glyph, but are "copied" from and linked to some other glyph. Thus whenever the original component contour is changed, all the composite glyphs that copy the component also change. The contour of composite components appears in dashed lines in the Glyph window.

Composites have the advantage of allowing the user to create only one instance of a component that is frequently found in a font and reusing it without having to redraw it each time. Later if the design of the component changes it need only be altered once — in the original component. And finally, a composite takes up less room in the font than an outline, allowing for smaller font files.

There are three operations related to composite glyphs: adding a component to glyphs, decomposing a component and positioning a component.

Adding a Component

To add a component to a glyph currently open in the Glyph window, select the **Add Component** command from the Glyph window context menu.

You will see a dialog box that is similar to the Find Glyph dialog box:

The only difference is that only those glyphs that can be used as component glyphs will appear. Of course, a glyph cannot be a component for itself.

A composite glyph can be used as a component glyph. It is automatically converted to source components.

Another difference is that you can set the position by entering its **X** (horizontal) and **Y** (vertical) coordinates. The component position is the distance between the composite zero point and the component's zero point.

To add a component you select the glyph you want to use as a component in this dialog box and press the **OK** button.

Another way to add a component is to drag it from the Font Window and drop it in the Glyph Window while the CTRL key is pressed.

Decomposing

To decompose a composite glyph, select the **Decompose** command from the **Glyph** menu or from the Glyph Window default context menu. The outlines of all components will be scaled and shifted according to their settings and added to the composite glyph. If the component glyphs had hints then these hints will also be added. The link to the original component will be lost.

To decompose an individual component in a composite glyph, right-click on the component and select **Decompose** in the context menu.

Component Positioning

To activate the component positioning operation, activate the Edit tool and click on the component's outline.

Alternately, if the current glyph is composite-only (so if does not have any "normal" outlines), press the **PAGE UP** and **PAGE DOWN** keys to select a component for editing.

You will see a control box surrounding the component with four corner handles, a cross in the center, a centerline and the number of the component in the components list.

To select another component, press the **PAGE UP** and **PAGE DOWN** keys or the **TAB** key.
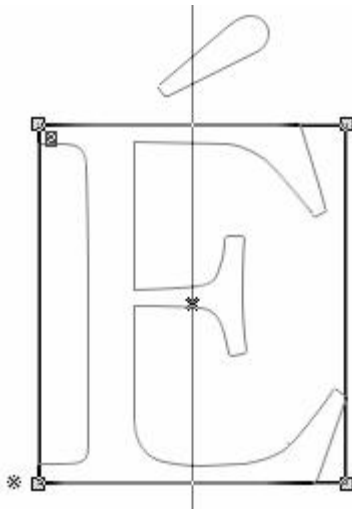
To select several components, click on each of them while holding down the **SHIFT** key.

To move the component position the mouse cursor inside the control box, hold down the left mouse button and drag the control box to a new location. If you position the cursor on the cross in the middle of the control box you can set the position of the component more precisely because the cross will snap to the guiding elements while moving.

You can also use the keyboard to move the component. Arrow keys move the component in one font-unit increments, the **SHIFT**+arrow keys increase movement to 10 units, and the **CTRL+ARROW** keys increase movement to 100 units.

To scale a component position the mouse cursor on one of the handles, hold down the left mouse button and drag the mouse to change the size of the component. Hold the **SHIFT** key to constrain the proportions of the component. Hold the **CTRL** key to scale around the component's center.

Some other useful commands are available in the context menu that appears if you right-click on the editing area while component tool is active:

Decompose	Decomposes (adds the outline to the composite glyph) the current component
Delete	Removes the component
Copy Metrics	Copies metrics data from the component to the composite glyph
Edit Component	Opens a new Glyph Window with the currently active component
Properties	Opens the Component Properties panel (described below).

Component Properties

You can **set the precise size and position of the component.** Right-click on the component with the Edit tool. You will see a context menu. Select the **Properties** command in this menu and you will see the Component Properties dialog box:
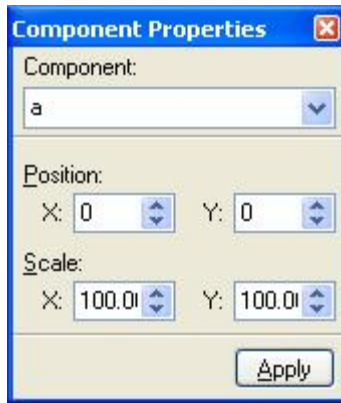
In this dialog box you can select a different glyph to be used as a component and set the component's position and scale. The component position is the distance between the composite zero point and the component's zero point.

➲ Tip: Double-click on the component to get the Component Properties panel.

Importing and Exporting Glyphs

With TypeTool you can exchange outline data with other vector-editing programs, either using the Clipboard or files. The most common format for vector data is Encapsulated PostScript (EPS).

Vector editing programs such as Adobe Illustrator and Macromedia Freehand typically are able to open and save EPS files. EPS was the native file format of Adobe Illustrator until version 8.0, though more recently, the Adobe Illustrator file format (.AI) is based on PDF rather than EPS.

TypeTool can exchange outline data with Adobe Illustrator via the Clipboard, and also export and import glyphs to and from AI-compatible EPS files. On one hand, you can use Adobe Illustrator or other compatible applications to draw your glyphs and then import them into TypeTool. On the other hand, files exported from TypeTool can be opened in any program that supports AI-compatible EPS files, e.g. Macromedia Freehand, Corel Draw, ACD Canvas etc.

By default, all font units in TypeTool correspond to points in Adobe Illustrator or other vector drawing applications. This means that if you want your uppercase letter H to be 700 units high in TypeTool, you should make it 700 pt high in Illustrator. 72 pt = 1 inch, so 700 pt = 9.72 inch.

Exporting Glyphs

To copy part of the glyph's outline to a vector-editing program use the usual copy-paste procedure. The selected portion of the outline will be copied to the Clipboard. Then switch to your vector-editing program and select the **Paste** command from the **Edit** menu.

To export a glyph to an Adobe Illustrator 8-compatible EPS file:

1. Select the **EPS** command from the **File > Export** menu.

2. Select the export directory and enter the name of the EPS/AI file in the standard File Save dialog box.

3. Press the **Save** button in the dialog box, and the EPS/AI file will be exported to the designated directory.

You can also export several glyphs at once: switch to the Font Window, select the glyphs that you want to export and select the **EPS** command from the **File > Export** menu. You will see a Save File dialog box where you enter a prefix file name for the exported glyphs. Each glyph will be exported to its own file with the file name consisting of the prefix plus the sequential number of the exported glyph.

Preparing Artwork in Adobe Illustrator

If you intend to use Adobe Illustrator to draw the glyph outlines:

In Illustrator, go to **Edit > Preferences > Units & Undo** or **Units & Display Performanc**e. Change all units to points (1 point is equal to 1 unit in TypeTool). Go to **Preferences > Files & Clipboard**. Disable **PDF**, enable **AICB** and select **Preserve Paths**. In **Preferences > Guides & Grid**, set **Gridline every: 10 pt** and **Subdivisions: 10**.

Still in Illustrator, select **File > New**. Set the width of the document in points to be the double of the UPM size of your font (e.g. 2000 pt for a 1000 UPM font). Set the height of the document to be the same as UPM size — Descender (e.g. 1000 — (-263) = 1263 pt). Select **Window > Info, View > Show Rulers**, **View > Snap to Grid**. Disable **View > Guides > Lock Guides**. Optionally select **View > Show Grid**.

Now click on the top ruler of the Illustrator document window and drag out a guideline. Position it at the height that has the same (positive) value as the (negative) descender of your font (e.g. 263 in our example). From the left ruler, drag a guideline and position it at 0. Click on at the top left corner of the Illustrator document window (where the top ruler and the left ruler meet) and drag out the origin point to where the two guidelines cross. Finally, click on the top ruler and drag guidelines to the positions of your ascender, x-height, and caps height.

You can draw your letters. Remember to assign some kind of fill to all your Illustrator drawings and avoid drawing letters that exceed the bottom or the top of the document size.

If you have already drawn some letters before, copy them to the newly created document, place and re-scale so that they fit between the guidelines you have drawn. Remember that all points of your letters should snap to the grid (otherwise TypeTool will round their position).

When you finished drawing your glyph in Illustrator, choose **Select > All, Edit > Copy** if you want to copy the outlines via clipboard or **File > Export > Illustrator Legacy EPS** or **File > Save As**, and select **Illustrator 8 EPS** as your file format, if you want to save the artwork as an EPS file.

Importing Glyphs

To paste an outline from a vector-editing program into TypeTool select the outline object that you want to copy and choose the **Copy** command from the **Edit** menu (in the source application). To place the copied outline in TypeTool switch to TypeTool (Glyph Window) and select the **Paste** command from the **Edit** menu.

To import an Illustrator 8-compatible EPS file into TypeTool, open a Glyph window (make a new glyph if necessary) and choose **Edit > Paste** if you are pasting from clipboard or **File > Import > EPS** if you are importing from a file.

Manual and Automatic Scaling

If the imported drawings end up being too large or too small, go back to your outline-drawing application and scale the artwork accordingly. Remember that if the option **Keep size** in **Tools > Options > General > EPS and bitmap background** is enabled, 1 pt in Illustrator/EPS corresponds to 1 font unit in TypeTool and artwork is imported without any scaling.

Alternatively, instead of scaling all your artwork to a particular height (e.g. 700 pt) in Illustrator, you can also have TypeTool automatically scale the artwork for you. This is particularly useful if you import pre-existing logos or similar symbols to TypeTool. If you wish that TypeTool automatically scales all pasted or imported artwork to fit the font's height, enable these options located in the **Tools > Options > General > EPS and bitmap background** dialog box:

⊙ Fit between Ascender and Descender

or

⊙ Fit between Ascender and Baseline

↘ Nodes in digital fonts can only have integer coordinates. On the other hand, your Illustrator artwork can have nodes with fractional coordinates such as 161.352 pt or 354.78 mm. When TypeTool imports a drawing, it has to round them to integer values — because it cannot generate fonts with fractional coordinates. The smaller your object is, the more extreme the rounding (and therefore, distortion) will occur. Therefore, we advise that you always scale your artwork in Illustrator to the appropriate size before copying it to TypeTool so that no rounding will be minimal. Also, if you work in Illustrator or similar applications, avoid fractional coordinates altogether by setting your grid to 1 pt and making sure that all your nodes snap to it.

↘ Note: Remember that TypeTool can only edit font outlines, not features such as color of outline, outline width or fill color. Regardless of the settings you have in the vector-editing application, only information about outlines will be copied to TypeTool. Ideally, in your vector-editing application set the fill color of all your objects to 100% black, and the width of the outlines to none.

Printing a Glyph

To print a sample of the current glyph, select the **Print** command in the **File** menu while the Glyph Window is active.

Refer to the "***Printing Glyph Sample*** (on page 345)" section for further details.

Editing Metrics

The tools in TypeTool for editing metrics data are common to all Fontlab applications, so if you have learned how to use these tools in TypeTool you will be ready to use these same tools in any of the Fontlab programs.

What are Font Metrics ?

A program that aligns and spaces text calculates the total width of all the glyphs in a paragraph. It then adjusts the widths of the space glyphs that separate the words and tries to put as many glyphs as possible into one line. The information about the words that are used to make a paragraph, and the information about the width of the individual glyphs is the only information necessary. To determine distances between lines, the application uses information common to all glyphs in the font, such as the length of ascenders and descenders, and a suggested line gap, and places the lines of text on the page using these distances. This information about horizontal and vertical spacing is what is known as font metrics.

All font and glyph metrics are expressed in font units, the same units that are used to measure node coordinates and settings such as UPM size.

There are four principal types of metric information in fonts:

1. Vertical font metrics (also known as *font family metrics*): metric values common for the entire font and often shared across a family, used to determine the linespacing. This includes baseline, the ascender and descender lines, the caps height, the x-height and the line gap. These are discussed in the "***Font Header*** (on page 313)" chapter and the "***Vertical Metrics*** (on page 238)" section of the "***Glyph Window*** (on page 139)" chapter.

2. Horizontal glyph metrics (usually referred to as *glyph metrics* or just *metrics*): metric values of individual glyphs that are used to compute line lengths. This includes advance widths and sidebearings. These are discussed in this chapter as well as in the "***Metrics*** (on page 233)" section of the "***Glyph Window*** (on page 139)" chapter.

3. Kerning: pair-wise adjustment of horizontal glyph metrics.

4. Vertical glyph metrics: the vertical advance "widths" (called *vertical advance vector*) for Asian glyphs used to type in vertical direction from top to bottom. These are discussed below.

This chapter discusses primarily the horizontal and vertical glyph metrics as well as kerning. For vertical font metrics, please consult the "***Font Header*** (on page 313)" chapter.

Horizontal Glyph Metrics

Each glyph in the font has a bounding box, a rectangle positioned in a theoretical rectangular cell. The most extreme nodes of the glyph determine the bounding box. Each glyph usually also has sidebearings: extra space to the left of bounding box (left sidebearing, LSB) and to the right (right sidebearing, RSB). The sum of the sidebearings and the bounding box width define the advance width (often just called *width*).

The intersection of the baseline and the left sidebearing is called the zero point. Horizontal (x) node coordinates to the right of the LSB line are positive and coordinates left of the LSB line are negative. Similarly, vertical (y) node coordinates above the baseline are positive and those below the baseline are negative.

When an application is laying a line, it positions the next glyph's LSB line right at the RSB line of the previous glyph.
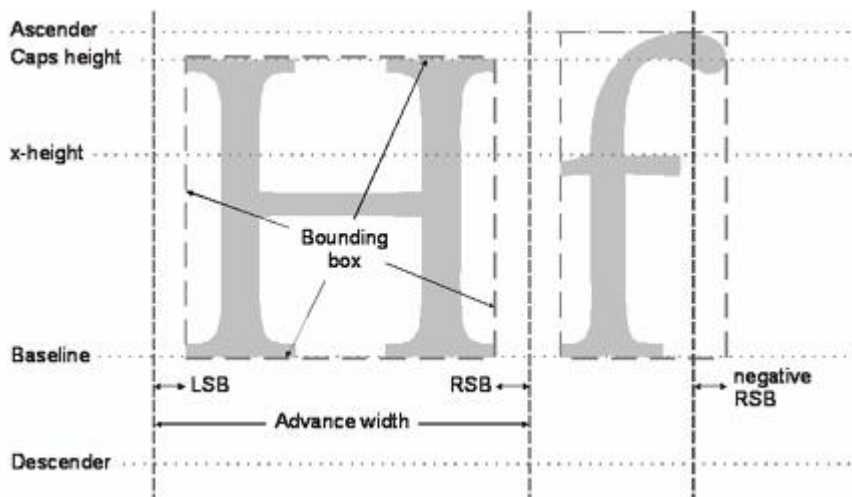
Glyphs may have negative sidebearings, e.g. the rightmost edge of the bounding box may be positioned to the right of the RSB line.

Kerning

Kerning information is used to adjust the space between specific pairs of glyphs. As you can see in the following image some glyphs may be well spaced with just the bearings rectangle but other glyphs are not. To fix this problem a special technique called kerning has been developed.

A good example is the "Av" pair. In the following image you can see two examples of inter-glyph spacing, with and without kerning:

With kerning   *Without kerning*
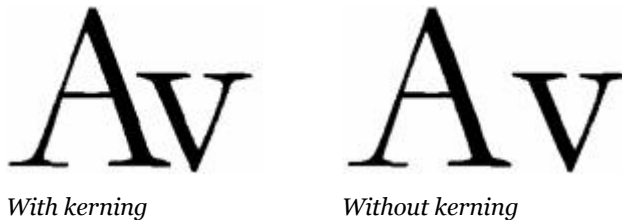
You can see that only the kerned image is optically correct because it can compensate for the problem caused by the special form of the "v" and "A" glyphs printed in sequence that leaves too much space between the letters.

Older font formats (Type 1, MM, TrueType without OpenType tables) implement kerning using kerning pair lists. Each kerning pair defines the number of font units (usually negative) by which the right sidebearing of the first glyph in a pair should be horizontally shifted when the glyph is followed by a specified second glyph. In the example above, the advance width of the "A" glyph may be 400 units and the advance width of the "v" glyph 250 units. The kerning pair "A v -50" defines that if "A" is followed by "v", the advance width of "A" should be reduced by 50 units.

A typical problem of the plain kerning pair list approach is that for accented glyphs, many duplicate pairs need to be included in the font. The pairs "Av", "Äv", "Áv" etc. usually should be kerned by the same amount, yet each of them needs to be included separately in the font — otherwise it will not be kerned. This results in rather large tables that unnecessary increase the size of the font and may hamper the performance of some applications. Therefore, in OpenType fonts, a more sophisticated kerning approach called class-based kerning has been developed to help address this problem.

Vertical Glyph Metrics

When typing text in some Asian languages, it is often necessary to specify the vertical alignment of glyphs in the text. In this case, information about the vertical glyph metrics is stored in the font file:
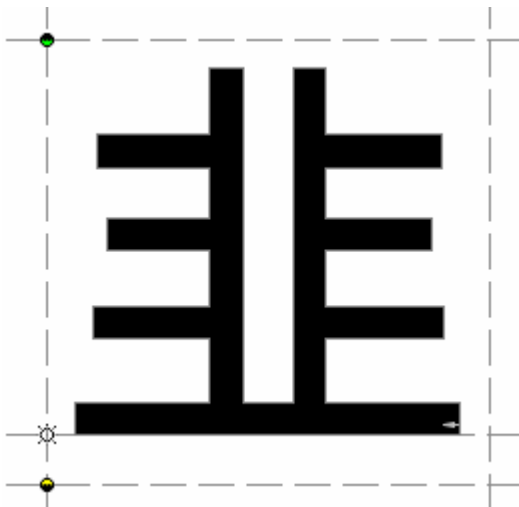
Usually, all Chinese, Japanese or Korean glyphs written in vertical layout have the same vertical advance "width" (called the *vertical advance vector*) so only the position of the glyph within the rectangular glyph cell needs to be specified.

However, it is possible to adjust the vertical advanced vector of individual glyphs. To define a vertical glyph advance vector in TypeTool, open the glyph in the Glyph Window, hold down the SHIFT key and move the base line. You will be able to set the top vertical glyph sidebearing (marked with a black-green symbol) and the bottom vertical glyph sidebearing (marked with a yellow-black symbol).

Note that this information is only used by applications that support vertical text layout, and only if the vertical glyph metric information is specified for all glyphs. Do not confuse vertical glyph metrics with vertical font metrics, i.e. ascender or descender lines that are used in text that is set horizontally.

Metrics Files

Information about the advance width of a glyph is usually located in font files. Kerning information may also be included in the file. In OpenType and FontLab font formats both metrics and kerning data are located in a single font file. In Type 1 (PostScript) fonts the metrics and kerning data are located in separate files.

There are two possible formats for the metrics files that are used with Type 1 fonts: AFM and PFM. AFM files (*Adobe Font Metrics*) are text files containing all the metrics and kerning information for a given font. These files are legible as text and can be edited in any text editor. PFM files (*Printer Font Metrics*) are metrics and kerning files used by the Windows operating system. They are binary files and cannot be read without special utilities. AFM files are a standard format for the exchange of metrics information for PostScript fonts. This information can be read directly by several operating systems and programs.

TypeTool can import and export metrics and kerning information in any of these formats.

Metrics Window

TypeTool has a special window where you can edit the metrics and kerning information. It is called the Metrics window.

To open the Metrics window select the **New Metrics Window** command in the **Window** menu. The Metrics window will appear:
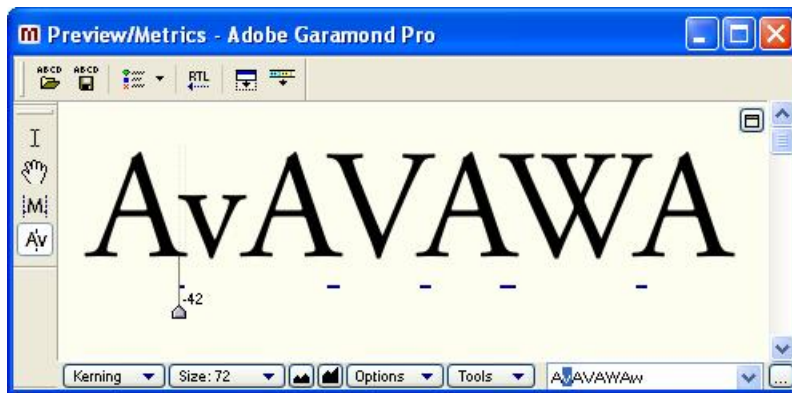
The Metrics window consists of several parts:

1.   A Metrics window toolbar with controls for importing and exporting metrics files, automating metrics or kerning generation and other commands:

By default the toolbar is docked to the top of the window, but you can drag it to the bottom or leave it floating around.

2. A Metrics Tools toolbar with four buttons that allow you to select one of the editing modes:

By default this toolbar is vertically aligned and docked to the left side of the window. You can drag it anywhere or dock to any side.

3. A local command area that is used to select a mode for the Metrics window and a string for metrics or kerning editing:

4. The editing area where the edited string with controls appears.

5. The header button, located in the top-right corner of the window:

Use this button to switch the local command area between top and bottom locations (see below).

The local command area of the Metrics window may be located in the bottom (default) or top area of the window. When the local command area is in the top location, it includes controls to modify metrics or kerning:
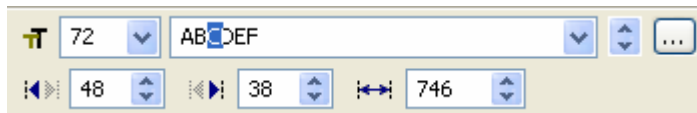
The content of this command area depends on the current mode of the Metrics window.

Editing Modes

The Metrics window may work in four different modes:

Text mode	Is used to enter and edit text in the main editing area. Works very similar to any standard text editor such as Notepad or TextEdit
Preview mode	This mode is used to preview text with kerning applied and check it at different sizes. Also the position and width of the underline and middle-stroke line can be adjusted in this mode
Metrics mode	This mode is used to adjust the metrics of individual glyphs. Kerning is not visible in the metrics mode
Kerning mode	In this mode you can edit pair kerning.

Other things that appear in the Metrics window are the: Ruler and Panel.

Metrics Ruler

The Metrics Ruler is a narrow bar located above the editing area:

| 743 | 660 | | 746 | 813 | 690 | 538 | |

Its purpose is very simple: to provide an overview of metrics and kerning data for the current line of text in the editing area.

The Metrics Ruler shows the advance width of the glyphs (in the middle of the glyph cell) and kerning. Kerning data appears on a light-blue background if kerning is negative (as in the AV pair) and on a yellow background when kerning is positive.

Of course, kerning information appears on the ruler only when the Metrics window is in kerning or preview mode.

The Metrics Ruler also may be used to create new global guidelines, but we will talk about that later.

You can control the appearance of the ruler using the **Ruler** command in the **Options** menu (if the local command area is at the bottom) or with the **Ruler** button on the Metrics Window toolbar.

Metrics Panel

The Metrics Panel is a horizontally oriented table that may appear above or below the editing area:

N ▸	H	A	M	B	U	R	G	E	V
↔	853	743	981	660	746	697	818	690	706
⊢	35	-1	8	31	8	52	48	46	28
⊣	37	0	31	64	7	-13	36	29	-19
Ke									-25

The Metrics Panel includes the following information for every glyph in the editing field: name, advance width, left and right sidebearing and a pair kerning value with the next glyph.

You may control the appearance of the Metrics Panel using the **Panel** command in the **Options** local menu (when the local command area is at the bottom) or with the **Panel** button on the Metrics window toolbar: 🖵.

Click on this button in the top-right area of the panel to move it top or bottom:
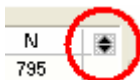
If you click on any cell in the Panel you may change the value:

Press the **UP** and **DOWN** arrow keys to navigate between different values for the same glyph. Press the **TAB** and **SHIFT+TAB** keys to navigate between glyphs.

When the Metrics Panel is visible, the properties area of the command area (if it is at the top) disappears.

Context Menu

As in all other windows of TypeTool, if you right-click on the editing area, you will see a context menu which contains commands that are related to the current mode of the Metrics window.

Every Metrics window mode has its own context menu:
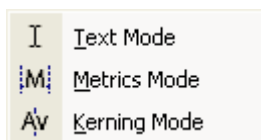
Preview mode context menu
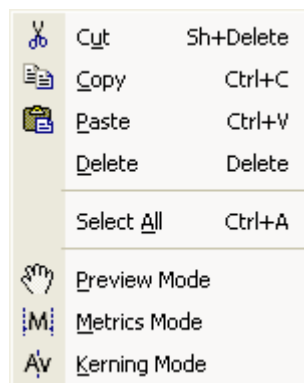
Text mode context menu
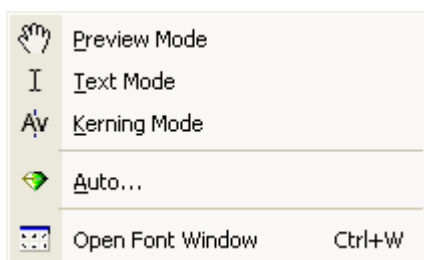
Metrics mode context menu
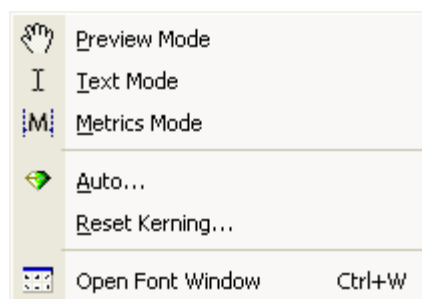
Kerning mode context menu

Metrics Window Toolbar

This is a simple list of all the buttons available on the toolbar:

AFM	Opens a metrics file (PFM, AFM or MMM format)
AFM	Saves a metrics file
▾	Opens a command menu (see below).
RTL	Changes the preview panel to the right-to-left reading mode
	Opens the Panel
	Opens the Ruler

The **Command** menu contains the following commands:

	Opens the Automatic Metrics or Automatic Kerning generation dialog boxes.
✕	Opens the Reset Kerning dialog box.

Selecting a String for Previewing or Editing

To prepare text for editing you have the following options:

1. Select one of the predefined sample strings in the sample text combo-box:
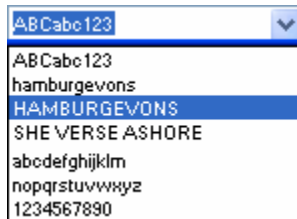
2. Enter the text in the sample text field of the control area (top or bottom located):
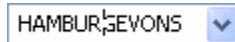
3. Enter the Text mode and type sample text directly in the Editing area.

4. Append glyphs to the sample text by dragging them from other windows.

Selecting a Predefined Sample String

Click on the button to the right of the sample text field and select the string for editing:
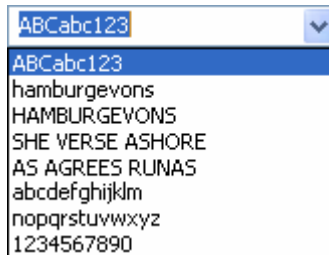
Or use the spin buttons to the right of the field to select the next or previous string:

You can also use the **CTRL+PGUP** and **CTRL+PGDN** keyboard shortcuts to navigate the list of sample strings up and down.

Editing a Sample String

Click on the sample string text field and modify it as you want. You may type text into it or you may use TypeTool glyph-access notation to access glyphs that have no characters mapped to the current keyboard layout.

TypeTool Sample Text notation:

Character	Meaning of the following text
/	Glyph name follows the slash: /A
	Follow the name with another '/' to continue entering glyph names or enter a *space* after the glyph name to continue entering ANSI characters:
	/Acaron/Adieresis BCDEF
	You may enter the code of the character according to the currently active encoding or a codepage:
	/128/130
	In this case the code number must contain only digits.
//	'/'
/#	Unicode codepoint of the glyph in hex format
	/#0446
****	Unicode codepoint of the glyph in hex format may be preceded with 'u'
	\0445\0448\u0446 BCDE
****	'\'
\n	Line break in the preview

Customizing the Sample String List

If you click on the button to the right of the sample list control, you will see the following dialog box:
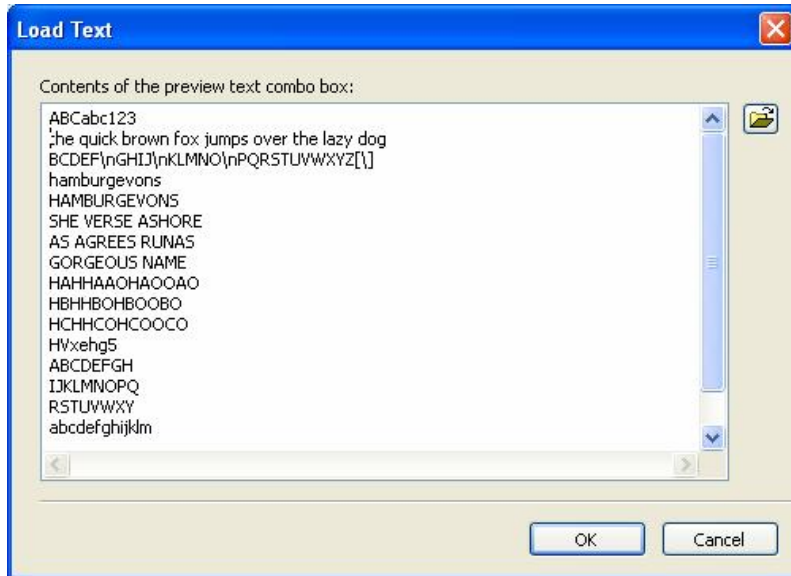
As you can see, there is a big multiline editing field that contains all the strings in the sample list. Change it as you want or click on the button to fill it from a text file.

You may use special characters as described in the previous section to enter glyph codes, names or Unicode codepoints. Type **\n** to force a line break in the sample text.

Enter some text, close the dialog box and then use the sample string scroll buttons or the **CTRL+PGUP** and **CTRL+PGDN** keyboard shortcuts to see how it works. Do not forget to click on the editing area before using any shortcuts.

Entering Text in Text Mode

You may edit text in the editing area similarly to how you do it in any text editor. Activate the Text tool on the Metrics Tools toolbar:
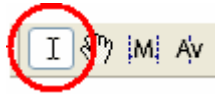
You can also select Text in the **Mode Selection** menu in the local command area docked to the bottom:
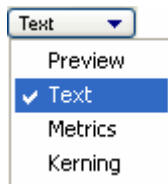
After the tool is activated you will see a caret cursor in the editing area. Start typing text. You may also drag-select text and use the **Edit > Copy** and **Edit > Paste** commands to move blocks of text inside the Metrics window or from external applications.

After the tool is activated you will see a caret cursor in the editing area. Start typing text
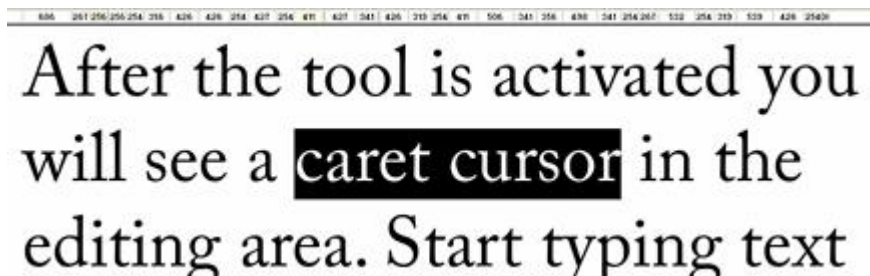
The copy-paste feature of the text tool is compatible with Unicode so if you paste some Unicode text, it will appear unchanged (if characters of that text are present in this font).

Check the sample string editing field, you will notice that it automatically creates TypeTool notation for all non-ANSI characters.

Using Drag-Drop

The easiest way to fill a sample string is **using the drag-drop method.**
You can simply drag any glyph from the Font Window and drop it in the
Metrics Window and it will be inserted in a position highlighted by the
caret. If you want to *add* glyphs to the string, hold down the **SHIFT** key. If
you want to *replace* the sample string with the dropped glyphs, hold down
the **CTRL** key.

Navigating in the Sample String

You can also use the **PAGE UP** and **PAGE DOWN** keys on the keyboard to
navigate within the sample string. The **HOME** and **END** keys will jump to the
beginning and end of the current line of text.

To scroll the window you can press the **SPACEBAR** and scroll the Metrics
window with the hand tool.

If the sample text is really long, switch the Metrics window into Preview
mode and use the hand tool to scroll the editing area.

Activating and Browsing Glyphs

Click on any glyph in the Editing area and it will be selected for further
editing. In metrics mode you will see the right and left handles that allow
you to change the sidebearings and in kerning mode you'll see a pair
handle that highlights a position between the first and second glyphs in the
pair.

After you activate a glyph you can browse the glyph collection in the
current font. Use the "previous glyph" and "next glyph" shortcuts. By
default they are **CTRL + [** and **CTRL +]** respectively.

In Metrics and Kerning modes you can change a glyph in the string by
pressing the related key on the keyboards or by quickly entering its name.

Selecting Preview Size

If the local command area is in the top part of the Metrics window, type the desired point size in the **String Size** combo box:
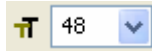
or select one of the predefined sizes from the list.

If the local command area is at the bottom, you will get a size menu:
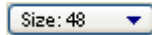
and two buttons to the right of it:

Use these buttons to decrease or increase the size of the sample string.

Both the combo box and the **Size** menu contain an **Auto** command. Select it and the size of the sample text will be automatically selected to fit one line of text into the current vertical size of the editing area.

The **Size** menu (available when the local command area is at the bottom) also has the **Custom**... command. Select it and you will see the following dialog box:
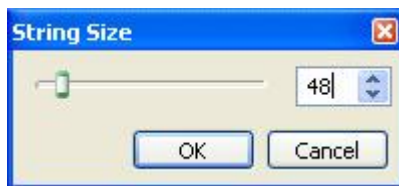
Type in the desired point size in the text field at the right or use the slider to adjust the size. You will see the result immediately in the Metrics window.

If the sample text becomes too large to fit in the window, a vertical scroll bar will appear allowing you to view all the editing areas of the Metrics Window.

Right-to-Left Mode

If you are working on a font that requires right-to-left reading, like Arabic or Hebrew, you can change the Metrics window to the right-to-left mode. Click on the ![RTL] button on the toolbar and you will see that the preview string is written from right to left:
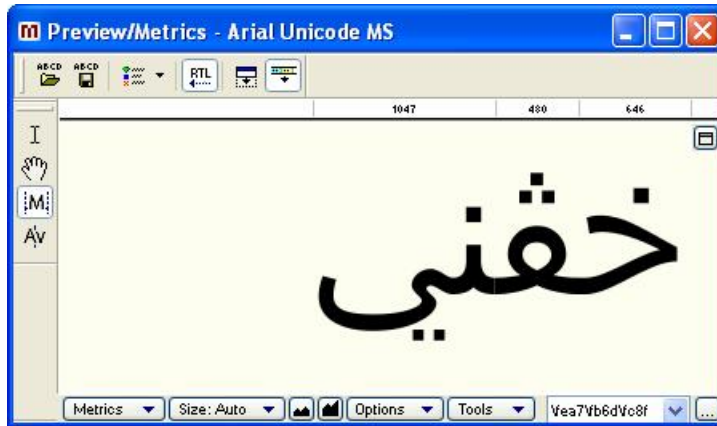
❧ Note: The Metrics window in TypeTool does not support OpenType Layout features, so Arabic shaping is not automatically performed. You need to explicitly enter the glyph names or Unicode codepoints of the presentational forms to display the text.

Previewing Outline and Nodes

Some commands in the **View > Show Layers** menu work when the Metrics window is active:

┼	**Guidelines**	Global guidelines are visible in the current line
∴∴	**Glyph metrics**	Baseline is visible
⚏	**Vertical metrics**	Font vertical metrics are visible in the current line
⟋	**Nodes**	Nodes are visible
◧	**Preview**	Outlines are filled

This means that it is not necessary to have the glyph outlines always filled while you are working with font metrics. For example, you may need to switch off the fill and switch on the nodes to visually compare placement of nodes in some glyphs:
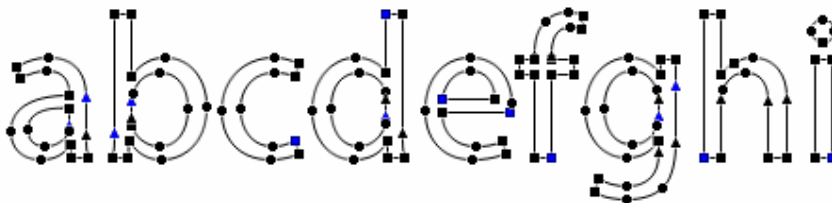
Customizing Colors

You are not restricted to black text on a white background. Open the **Metrics options** page (Tools > Options > Metrics Window):
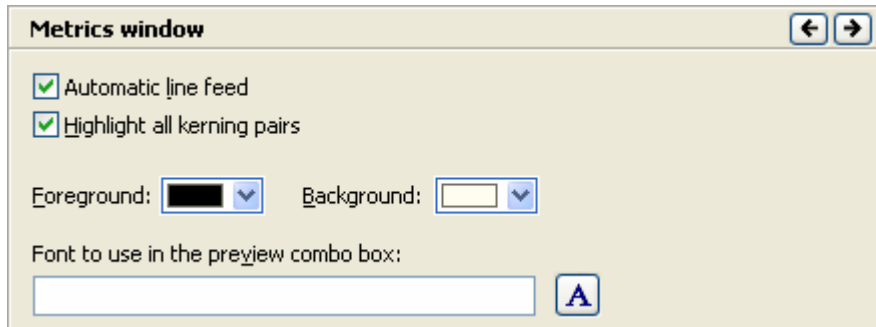
Use these controls to customize the foreground and background colors:

As a result, you can get custom colors in the Metrics window:

Editing Underline and Strikethrough

To edit the position and width of the underline and strikethrough line, switch the Metrics window to the Preview mode. You can do this by clicking on the **Preview Mode** button 🖐 of the Metrics Tools toolbar or by selecting **Preview** in the mode-selection menu of the bottom command area or by selecting the same command in the context menu that appears when you right-click on the editing field.

To get access to the controls for adjusting the properties of the lines, switch the local command area to the top using this button: ▣ (located in the top-right corner of the editing area). This is how the Metrics window should look:
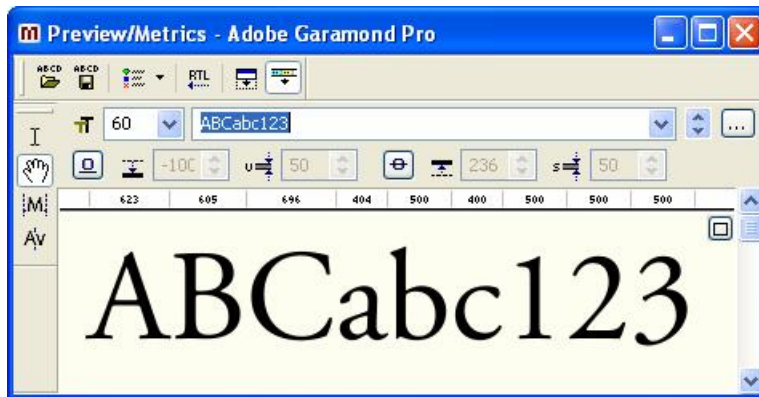
Above the ruler you can see the lines controls:

There are two buttons and four editing boxes. Click on the **Underline** button to show the underline:
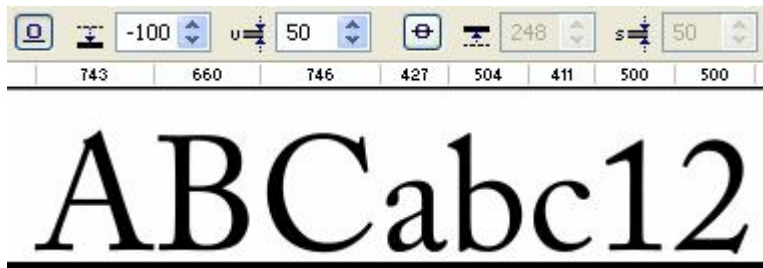
As you can see, underline controls are now enabled so you can use this control: to change the underline position. Use this control: to change the underline thickness.

Use the **Strikethrough** button to show the strikethrough line and enable the related controls:
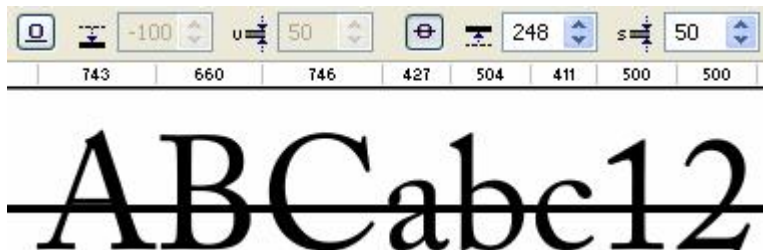
Strikethrough may appear together with underline or separately (as imaged).

Editing Metrics

This section discusses horizontal glyph metrics (the advance width and the sidebearings, jointly referred to as just *metrics*), and kerning. In TypeTool you can modify this information either manually or automatically.

Horizontal glyph metrics can be modified manually in the Glyph window by dragging on the sidebearing lines. However, this does not give you an accurate presentation of the glyphs in context. The process of letterspacing (devising of glyph metrics and kerning) should not be done for each individual glyph separately. Inter-glyph whitespace should be designed based on words and strings of text. You can do this in the Metrics Window.

To modify glyph metrics, switch the Metrics Window to the Metrics mode: click on the **Metrics Mode** button ⌷M⌷ in the Metrics Tools toolbar or select the **Metrics** command in mode-selection menu on the bottom command area:
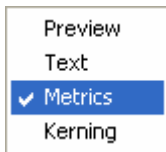
You can also right-click on the editing area and select the **Metrics Mode** command in the context menu.

The easiest way to see the metrics of a glyph is to use the Property area:
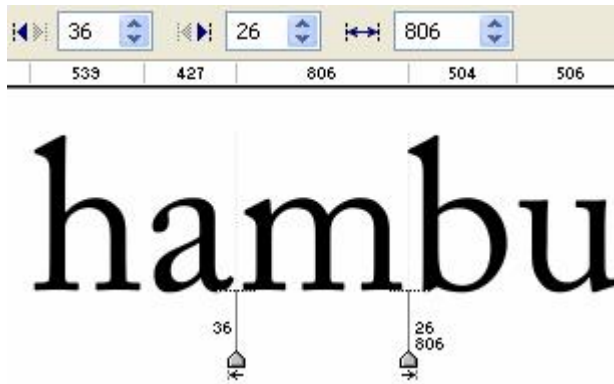
By default, the Property area is empty. To make the metrics editing controls visible, click the left mouse button on a glyph in the editing field. The metrics editing controls will appear and the sidebearings lines with editing handles will appear at the sides of the glyph.

The numbers at the bottom of the glyph are the left and right sidebearing values and the glyph's advance width.

Manual Metrics Editing

To modify a glyph's metrics you can use several methods:

1. Drag the sidebearings lines.

2. Drag the glyph within the editing area.

3. Edit the values in the property area of the Metrics Window.

4. Use the Metrics Panel.

To drag the sidebearings lines, position the mouse cursor on the line, hold down the left mouse button and drag the mouse. Release the left mouse button when you are done.

To drag a glyph within the editing area, position the mouse cursor on the glyphs' image; hold down the left mouse button and drag the mouse to position the glyph inside its advance width. Press and hold down the right mouse button while dragging the mouse to modify the glyph's advance width.

You can also **modify the vertical position of the glyph** relative to its baseline. Hold down the SHIFT key on the keyboard while dragging the glyph.

Using the Keyboard

When the glyph is active you can use the keyboard to adjust the metrics:

Left and right arrow keys	Moves the glyph by one font unit inside the sidebearings without changing the advance width. Hold down the **SHIFT** key to move the glyph by 10 font units
Ctrl+left and right arrow keys	Moves the glyph together with the right sidebearing. This changes the left sidebearing and the advance width. Hold down the **SHIFT** key to move by 10 font units at each keystroke
Page Up	Moves to the previous glyph in the sample line
Page Down	Moves to the next glyph in the sample line
Any character or digit	Selects the character you have typed as the current character for editing. You can also enter the glyph name if you want to access glyphs that are not assigned to any key combination
Ctrl+] and Ctrl+[	Moves to the next and previous glyphs in the font.

Using the Metrics Panel

Click on the ⬙ button to show the Metrics Panel:
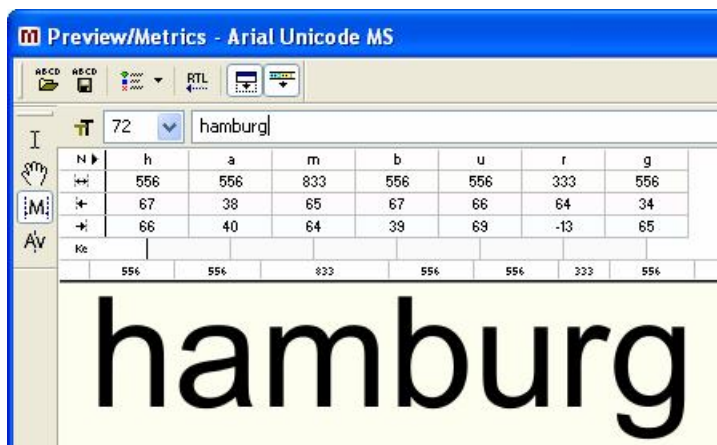
The Panel always consists of four lines:

- Glyph name
- Width
- Left sidebearing
- Right sidebearing

Click on any number in the panel to enter an exact value. Use the keyboard to adjust the number and press the **ENTER** key when you are done. The **ESC** key or a left-click outside the cell you are editing will cancel the changes.

Press the **UP** and **DOWN** arrow keys on the keyboard to move up and down in the panel. Press **TAB** key to move right and **SHIFT+TAB** to move left.

Referencing Metrics

In the Metrics Panel, you can use glyphnames as a reference instead of the real numeric values. For example, if you want to set the left sidebearing of the glyph 'B' to be equal to the left sidebearing of the glyph 'D', click on the cell located at the intersection of the B column and the third row:

and, instead of the numeric value for the metric, enter "=D". When you press the **ENTER** key to accept changes, the data will be copied from the source glyph.

Using the Calculator

TypeTool has very simple calculator embedded in most editing fields that allow you to enter formulas. Instead of entering a value you can enter an equation:

650/2

Which will produce 325 — the value that appears in the editing field. This calculator works in the Metrics Panel.

The standard 4 operations: + — / and * are accepted.

Automatic Metrics Generation

TypeTool can automatically define glyph metrics using a special algorithm. This algorithm usually produces good results but we recommend manual editing for the best results.

To automatically generate glyph metrics switch the window to the Metrics mode and select the ⬦ **Auto** command in the ▦ **Tools** local menu of the Metrics Window or in the context menu.

The Automatic Metrics Generation dialog box appears:
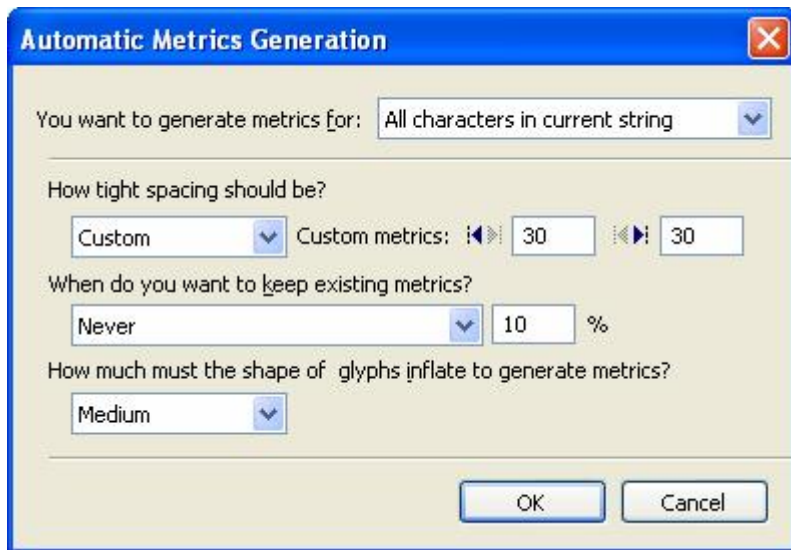
This dialog box includes two areas: **Area of application** and **Parameters**. In the first area you select the glyph(s) to which the automatic algorithm will be applied.

The possible choices are:

Current character only	This option is the default if any glyph is selected in the editing area
All glyphs in the current string	This option generates metrics for all glyphs in the current string in the editing area
Whole font	This forces TypeTool to generate metrics for all glyphs in the font and is not generally recommended. This operation is not undoable.

You can choose the parameters for the algorithm in the **Parameters** area of the Autometrics dialog box. All the parameters are displayed. We recommend that you experiment with various parameters using the autometrics application.

Editing Kerning

To edit kerning data switch the Metrics Window to Kerning mode by clicking on the ^Av^ button on the Metrics Tools toolbar.

or, select the **Kerning** command in the context menu that appears if you click the right mouse button in the editing area of the Metrics Window.

When you switch to the kerning mode and the metrics property panel is visible you will see the total number of defined kerning pairs for the current font appears in the property area of the Metrics Window:

Pairs #: 781

To make the Kerning Editing controls visible you must select the pair that you want to edit. Position the mouse cursor on the right glyph of the pair and click the left mouse button.

You will see the Kerning Editing controls appear in the property area and the kerning line and handle appear in the editing area:
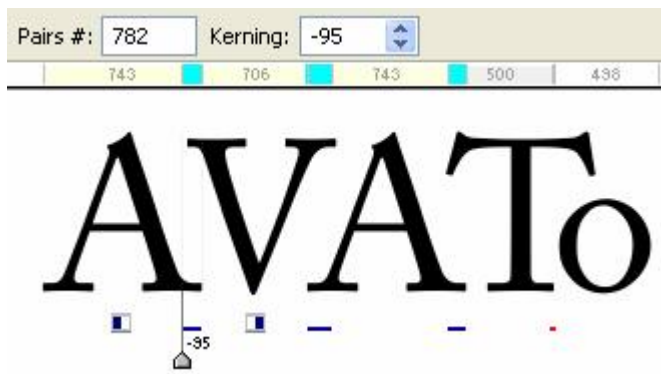
There is now a blue area in the metrics ruler. This means that negative kerning exists for that pair in the current preview string. If that area is bright yellow, it means that kerning between the two glyphs is positive.

Manual Kerning Editing

To edit kerning manually, drag the kerning line (or right glyph of the kerning pair) using the left mouse button. If you click the right mouse button while holding down the left mouse button on the glyph or on the kerning line, that **kerning pair will be removed.** You will see that the total number of kerning pairs decreases.

➲ Tip: if you hold down the ALT key and double-click the right glyph of the pair, it will be copied to the left of the left glyph:
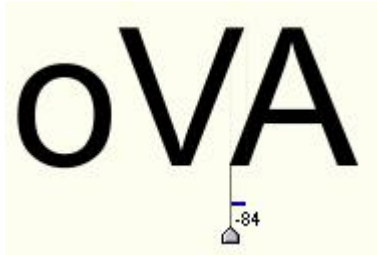
before ALT-double click on 'A'

after ALT-double-click on 'A'

Using the Keyboard

When a glyph is selected in the sample string you can use the left and right arrow keys to change the kerning by one font unit at each keystroke. Hold down the **SHIFT** key to change the kerning by 10 font units.

Press the **CTRL+[** and **CTRL+]** keys to change the glyph in the string and **PAGE UP** and **PAGE DOWN** keys to move to the previous and next glyph in the string.

Using the Metrics Panel

When the properties area is expanded and Kerning editing mode is activated you can see all the glyph metrics and pair kerning information in the editing field:

Kerning is displayed on the fifth row in the Metrics Panel and each value is positioned between the glyphs that form the kerning pair. The background color for the kerning value is white when there is no kerning, light blue if kerning is negative (glyphs are shifted toward each other) and yellow if kerning is positive.

To change the kerning value, click on the kerning row in the table and enter the new value. Press the **ENTER** key on the keyboard to accept the changes or **ESC** to cancel. Press the **TAB** and **SHIFT+TAB** keys to select a pair in the sample string.

Automatic Kerning Generation

The easiest way to apply kerning to a font is to use TypeTool's autokerning algorithm. This algorithm analyzes the shape of the glyphs in the given pairs and automatically kerns them. You can control the pairs list that the autokerning algorithm processes as well as other parameters.

To define kerning automatically switch the window to the Kerning mode and select the ♦ **Auto** command in the ⁂ ▾ **Tools** local menu of the Metrics Window or in the context menu.

The Automatic Kerning Generation dialog box appears:
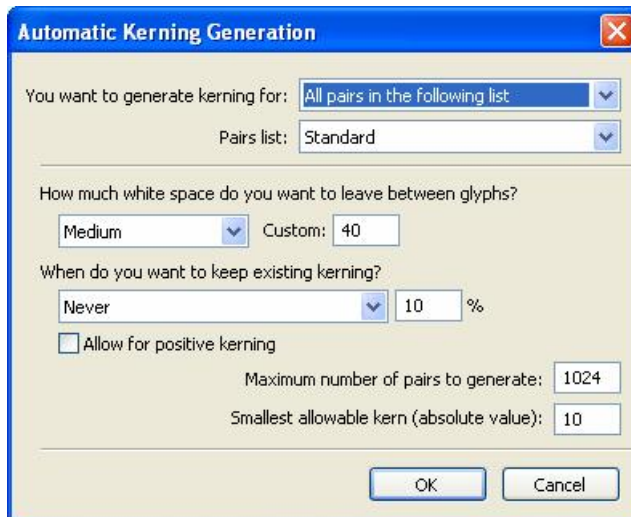
This dialog box consists of two areas: the **Area of application** and **Parameters**.

In the first area you select the pairs for which the algorithm will compute kerning values. You can choose between **Current pair only** (available if one of the pairs is selected in the editing area), **All Pairs in the current string,** or **All Pairs in the following list**.

TypeTool allows you to generate kerning for all the pairs located in a special list file. The list files are stored in [Application default data folder]\Kerning folder. You can create your own kerning pair files or use one of the files placed there at the time of TypeTool's installation.

The **Parameters** area lets you customise the autokerning algorithm. The most used option is: **How much white space do you want to leave between glyphs?** This controls how close the glyphs will be moved together while computing kerning in the pair.

The **Allow for positive kerning** check box lets the autokerning algorithm produce positive kerning in pairs. Positive kerning moves glyphs apart from each other. Positive kerning is usually not recommended but there may be occasional circumstances where it is needed.

If you want to save the existing kerning the combo box lets you control the disposition of the existing (imported or manually created) kerning pairs. You can replace existing pairs by automatically generating new ones, keep them unchanged, or select the condition mode.

The **Maximum number of generated pairs** and **Smallest allowable kern** options control the possible number of automatically created pairs and the minimal normal (negative or positive) kerning value.

Resetting Kerning

To remove the kerning information for some glyphs or for the entire font you must use the *Reset Kerning* feature. To open the Reset Kerning dialog box select the **Reset Kerning** command in the 📋 ▾ **Tools** local menu of the Metrics Window or in the context menu.

The Reset Kerning dialog box appears:

This dialog box includes options that control kerning removal.

Available options are:

Reset kerning for the current pair	This is the default if a pair is selected. Removes kerning for that pair only. You can get the same result by clicking the right mouse button while editing kerning in the current pair
Reset kerning for all pairs in the string	Default if no pairs are selected. Removes kerning in all pairs that exist in the current string
Reset kerning for all glyphs in the string	Removes kerning in all pairs that include glyphs in the current string
Completely reset kerning in the current font	Removes all the kerning pairs available in the font. Because this is not an undoable operation, we recommend you to save the current metrics and kerning data in the metrics file.

Opening Metrics Files

TypeTool allows you to import metrics and/or kerning information into the current font. Using this feature, you can create metric and kerning information once and use it in several similar fonts.

To import a metrics file into TypeTool click on the 🖻 button on the toolbar. You will see the standard Windows Open File dialog box. Select the metrics file that you want to import (in PFM or AFM format) and press the **Open** button.

The Import Metrics dialog box appears:
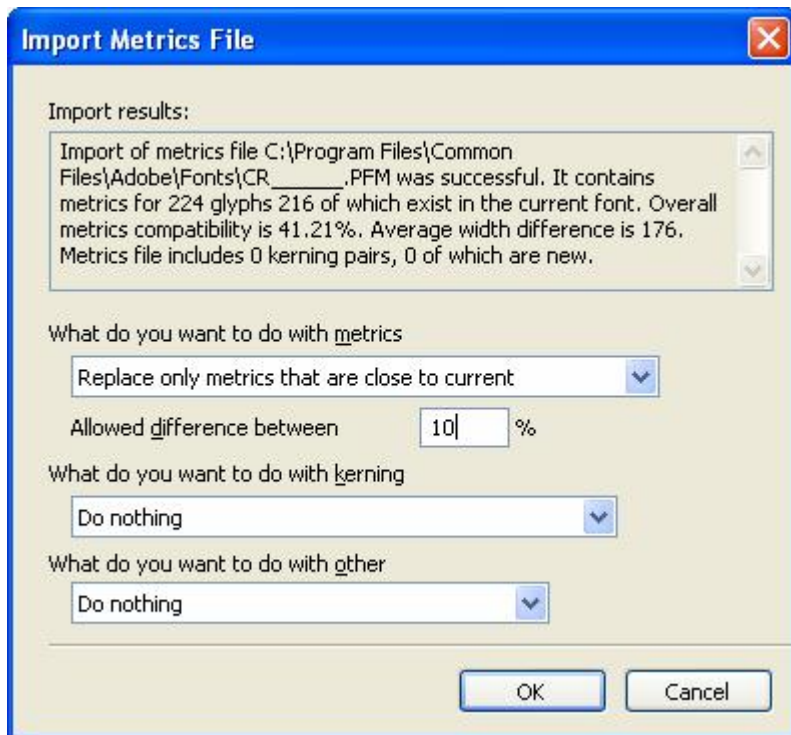
The topmost control contains a legend describing the metrics file that you are importing and its compatibility with the current font.

The options in the **Parameters** area let you select various metrics importing options:

What do you want to do with the metrics data:

Do nothing	Do not import metrics data from this file
Replace all metrics in the current font	Import all metrics data (glyphs' advance widths and sidebearings) and replace the metrics data in the current font. We recommend that you use this option only if your font is very similar to the metrics file that you are importing
Replace all metrics that are close to current	Replace only those metrics records that are similar to the imported metrics. The **Possible difference between metrics** option controls the allowed difference
Replace metrics that are thinner than in the current font	These options are obvious.
Replace metrics that are wider than in the current font	

What do you want to do with the kerning data:

Do nothing	Do not import kerning data from the metrics file
Completely replace kerning data in the current font	Remove all existing kerning pairs and replace them with pairs imported from the metrics file
Add imported kerning data to the current font	Leave the existing kerning pairs unchanged but add new kerning pairs from the metrics file
Add new kerning pairs but autokern them	Import information about the glyphs that form each kerning pair in the metrics file and apply an autokerning algorithm to these pairs.

The **What do you want to do with other data?** option controls the font header importing option. TypeTool can import the Font Info data from the metrics file and replace the current font info data if the **Replace this data in the current font** option is selected.

Saving Metrics Files

When you export a font file in Type 1 font format the metrics files (in AFM and PFM formats) are automatically written. The TrueType font format includes all metrics information so it is not necessary to export additional files.

However, if you want to export a metrics file alone, you can always do so by using the Metrics Window. Press the button on the Metrics Window toolbar. The Standard Save File dialog box appears.
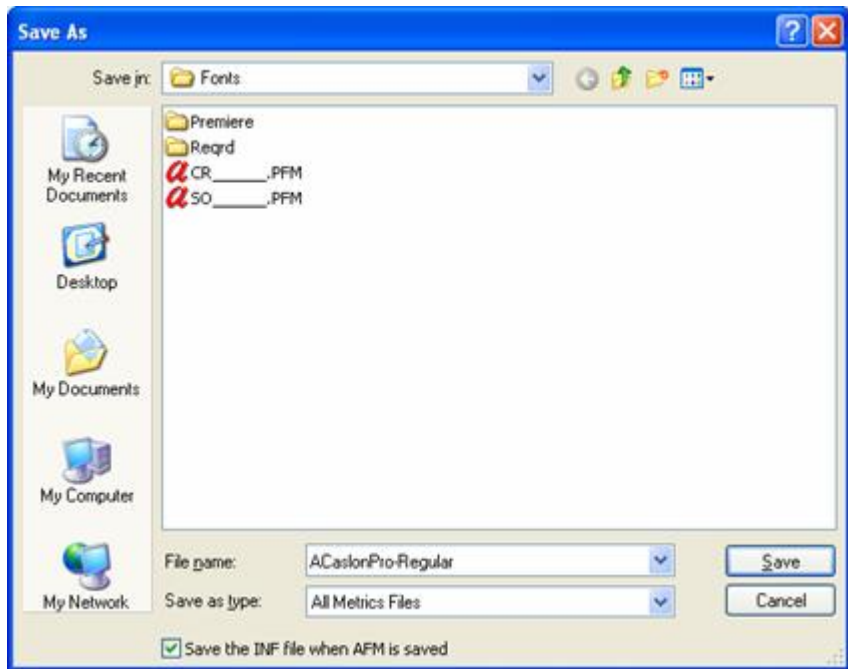
Select the destination format (AFM or PFM), and the destination directory. Enter the file name and press the **Save** button to save the metrics file.

You may choose whether to save the font information (.inf) file along with the .afm metrics file or not.

Printing Metrics

While you are in the Metrics Window you can print sample strings with or without metrics and kerning information. To do so select the **Print** command in the **File** menu.

Choose the **Font Sample** page:
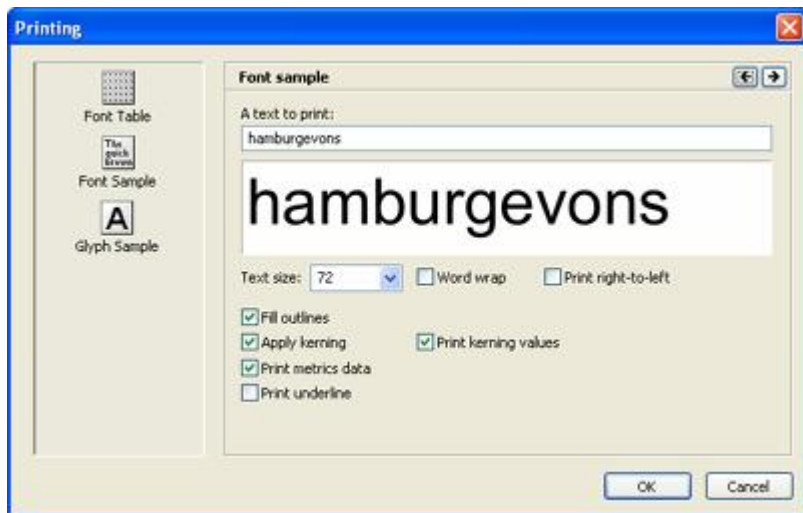
You will see that the "A text to print" field is populated with the sample string from the Metrics window. Select other appropriate options and click on **OK** to print the sample.

Check the "***Printing and Proofing Fonts*** (on page 339)" chapter for more printing options.

Actions

In TypeTool you can transform glyphs in many ways. You can edit glyphs and their metrics manually using the Glyph and Metrics windows described in previous chapters. Or you can use TypeTool's actions to edit glyphs or metrics automatically. Actions may be applied to one glyph, to a range of glyphs selected in the Font window, to the entire font or even to a large number of fonts. The actions allow you to apply various transformations to the glyph contours, change the font's metrics and kerning, modify hints and guidelines. Some automated actions — such as making glyphs bolder — are no replacement for a treatment professional type designer, but they usually give you a reasonable start. Other actions produce high-quality results that do not require manual control or correction.

In this chapter we will show you how to use the actions and give a detailed description of each available action.

The Actions Dialog Box

The easiest way to apply actions is to use the Actions dialog box. It is accessible from the **Tools** menu while the Font or Glyph window is active. Select the **Action** command from the **Tools** menu and you will see a dialog box:
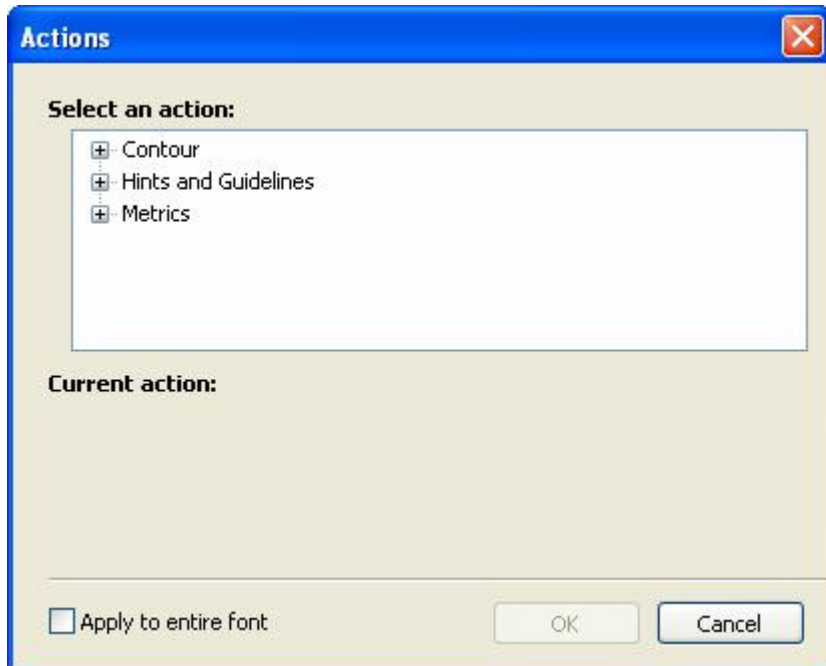
➘ Note: in previous versions of TypeTool, this dialog box was called Transformation.

If you open this dialog while the Glyph window is active the action will be applied only to the glyph currently open. If you open it while the Font window is active then the action will be applied to all selected glyphs.

To choose an action to run, use the list of actions:

Select an action:

田 Contour
田 Hints and Guidelines
田 Metrics
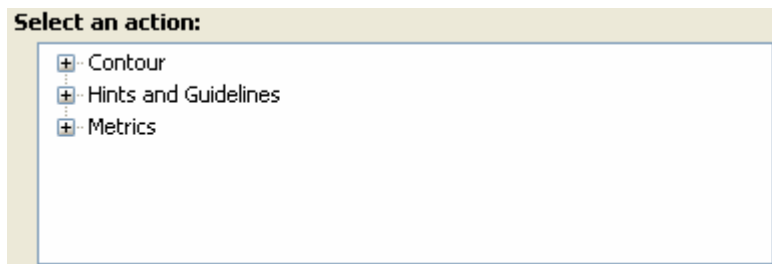
Expand one of the categories to see all the actions:

白 Contour
　　　 Mirror [H]
　　　 Contour direction [T1]
　　　 Rotate (15.00?< CW) Z
　　　 Scale (80.00, 100.00) HMZ
　　　 Shift (0, 0)
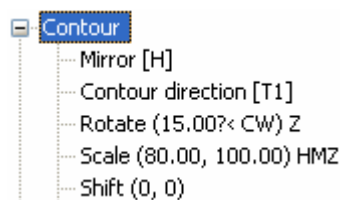
Some action names are followed by their parameters in brackets.

Select an action and you will see a parameter panel appear below the list:

Current action: Rotate glyph

Rotation angle: 15　　⊙ Clockwise　 ○ Counterclockwise

Rotate around: Origin point (0,0)

Rotation center: 0　 : 0
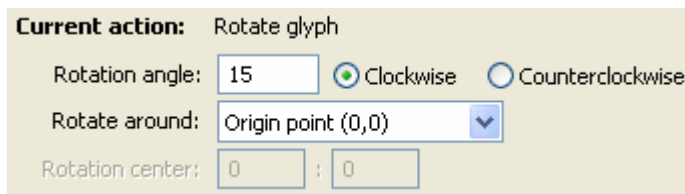
The contents of the parameter panel depend on the action selected.

After you select an action and set its options, press the **OK** button to run the action. If you are applying the action to a lot of glyphs a warning message will appear telling you how many glyphs you will modify and asking you for confirmation. Action applied to many glyphs is not undoable, so it is a good idea to save your font before running this action.

You can repeat the last action by choosing the **Tools > Repeat Action** command or by pressing the 🔳 button in the Transformation panel.

Below you will find a detailed description of each available action.

Actions

There are three groups of actions:

Contour	The outline of a glyph is transformed
Hints and Guidelines	Actions that are concerned with hints and links
Metrics	Metrics information is transformed (includes automatic metrics generation)

The actions in the Actions dialog can be applied to entire glyphs only, not to parts of glyphs. If you wish to scale, rotate or mirror a selected portion of a glyph rather than the entire glyph, please use the Transformation panel (**Windows** menu) or the Free Transform operation (context menu of an outline selection).

Contour Transformation

Here is a list of all the outline transformation actions:

Shift	Shifts the glyph's outline
Mirror	Mirrors the glyph vertically or horizontally
Scale	Scales the glyph proportionally or non-proportionally
Rotate	Rotates the glyph
Slant	Slants the glyph
Contour Direction	Sets the direction of contours to PostScript or TrueType or reverses all contours

Shift

Current action: Shift glyph

Horizontal shift: 0   Vertical shift: 0

☐ Shift the Mask layer

This action shifts the outline of the glyph in the vertical and/or horizontal direction. Here is a sample of a font with some glyphs shifted in the vertical direction:

Sample text

You can also shift glyphs in the vertical direction in the metrics mode of the Metrics window: hold down the **SHIFT** key and drag the glyph.

Use the **Shift the Mask Layer** control to shift the mask layer together with the outline or to leave it untouched.

Mirror

Current action: Mirror transformation

☑ Horizontal mirror ☐ Vertical mirror
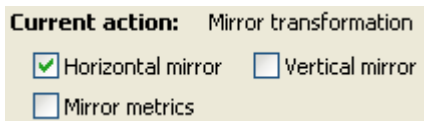
☐ Mirror metrics

Here is the result of this simple transformation:

The letters in the word "Sample" were mirrored horizontally and the letters in the word "text" — vertically.

Use the **Mirror Metrics** control to swap left and right sidebearings of a glyph.

Scale

Current action:	Scale glyph
Horizontal scale:	80 ☐ Proportional scale
Vertical scale:	100 ☑ Scale hints ☑ Scale metrics
Scale center:	Origin point (0,0) ▾

This action lets you scale your glyphs proportionally or non-proportionally. Enter your desired vertical and horizontal scale factors in the edit fields. Switch on the **Proportional scale** option to keep the vertical and horizontal scale factors the same.

Switch off the **Scale hints** option to avoid scaling hints along with the glyph's outline. Scaling hints' width is not always precise so if you scale hints with the outline you sometimes find that some hints now miss the nodes that they were supposed to hint.

Here is an example of this transformation (the letters of the word "Sample" were scaled 80% horizontally and the letters of word "text" were proportionally scaled to 120% of original size):

Sample text

Rotate

Current action: Rotate glyph

Rotation angle: | 15 | ◉ Clockwise   ○ Counterclockwise

Rotate around: | Origin point (0,0) ▾

Rotation center: | 0 | ; | 0

This transformation action simply rotates glyphs. You can set the rotation angle, the position of the center of rotation and the direction of rotation.

You can rotate glyphs around the origin point, around the reference point, around the center of the glyph's bounding box or you can specify a point that will be used as the center of rotation.

To specify the reference point, drag the glyph's origin point ☼ in the Glyph window.

Here is an example of the same rotation transformation around different center points:

Slant

Current action: Slant glyph

Slant angle: `12` ⦿ Slant to the right ◯ Slant to the left

Slant center: `Origin point (0,0)` ▾

This action slants glyphs. It is the quickest way to make an oblique version of your font. Apply this transformation to all the font's glyphs and correct the Font Info settings to let the operating system know that this font is now oblique.

Here is a sample of Slant transformation ("Sample" is slanted 12 degrees to the right and "text" is slanted 30 degrees left):

*Sample*text

Contour Direction

⦿ Set direction to PostScript (black on the left)
◯ Set direction to TrueType (black on the right)
◯ Reverse all contours

This action automatically detects the direction of contours and corrects them according to the option selected (PostScript if you're working in PostScript curves and intend to generate the font as Multiple Master, Type 1 or OpenType PS, or TrueType if you're working in TrueType curves and intend to generate the font as TrueType or OpenType TT). The **Reverse all contours** option changes the direction of all contours to the opposite.

Hints and Guidelines Transformation

Hints transformation actions let you automate hinting actions.

Remove hints/guides

☑ Remove horizontal hints   ☐ Remove horizontal guidelines
☑ Remove vertical hints   ☐ Remove vertical guidelines
☑ Also remove links

Use this action to remove hints and links or guidelines in selected glyphs. This command is the equivalent of the **Remove Hints** and **Remove Guides** commands that were described in the "*Glyph Window* (on page 139)" chapter.

Metrics Transformation

These transformations let you automatically increase or decrease a glyph's sidebearings and advance width.

Set Sidebearings

Use this action to change the sidebearings' values of the glyphs. You can set new values for the left or right sidebearings or change these values by entering the amount in font units. So if you think that your font needs some more white space, select this action, choose the **Increase by** option in the list boxes and enter the value by which you want to increase the sidebearings.

If the **Affect composites** option is off the action will not be applied to composite glyphs.

Here is a sample of increased glyph advance widths:

Font Header

Perhaps the most important information you need to define for a font is its header or font info data. This information is mainly used to properly register the font in the operating system and in any program that uses it.

It is very important to carefully define all font parameters. Even the best-designed font is useless if it cannot be installed.

Font Info Dialog Box

The control center where you define font parameters is called the Font Info Dialog box and is accessible from the **File** menu:
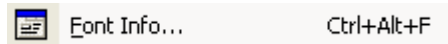
or with the button on any Font window:

The Font Info dialog box consists of three parts:
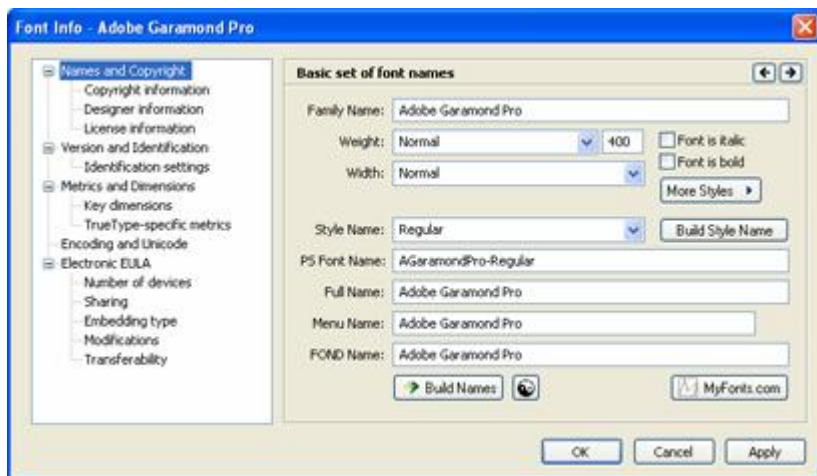
At the left there is a page selection control where you can choose one of the sections in order to edit part of the Font Info information:

When you select one of the pages, it immediately appears to the right of the list:
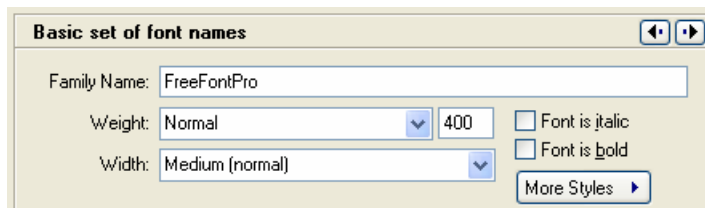
Use the arrow buttons in the top-right area of the page to browse all available pages:

Alternately you may use the **CTRL**+**TAB** and **CTRL**+**SHIFT**+**TAB** key combinations to browse pages.

Command Bar

In the bottom area of the dialog box you find the command bar:
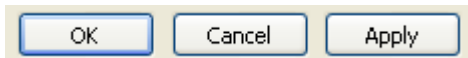
Here is the description of the buttons:

OK	Accepts changes you made to font info and closes the dialog box
Cancel	Cancels any changes and closes the dialog box
Apply	Accepts changes but lets you continue, so you can see the results of your changes in the Font, Glyph or Metrics windows.

Font Names

Names and Copyright
  Copyright information
  Designer information
  License information

The names section includes the most important font-registration information.

All programs use the information on this page to refer to a font. Be sure to enter all the values very carefully and use the automatic features where available.

Basic Identification and Names

Family Name	[name: 1] (this mark is the ID of the name in the TrueType and OpenType specification: ***http://www.microsoft.com/typography/otspec/name.htm*** (http://www.microsoft.com/OpenType/OTSpec/name.htm)) The name of the typeface to which the font belongs. All fonts that are from the same typeface must have the same **Family Name** field. The Family Name is used as the root of the Font Name so we recommend that you fill in this field first
Weight	Weight of the font. You may enter a custom value in this field or select one of the predefined weight names in the box. Values in this list are sorted by increased weight value. Choose **Normal** or leave this field empty if you do not care about the font's weight
Weight Value	Numeric weight value of the font. This number defines the font weight and is used by the operating systems to organize fonts to font families. TypeTool will fill it automatically when you select some Weight in the combo box, but if you want you can customize it
Width	The average advance width of the font's glyphs. Enter a custom value or select one of the predefined width values from the drop-down list. Leave this field empty or select **Normal** width if you do not care about the font's width
Font is Italic	[OS/2: fsSelection] Switch on this check box if you are creating an italic font

Font is bold	[OS/2: fsSelection] The Font is defined as bold. Usually this check box is related to the Weight setting, but it is not required. For example, if you are making a family containing Light and Normal styles, you may need to mark the Normal style as Bold so you will not need to split these styles into two separate families
More styles	Press this button to open a context menu where you can select one of the additional font styles. Only TrueType fonts use this information, but we recommend you always set it properly to simplify future font identification
Style Name	[name: 2] Contains complete style information about the font. We recommend that you fill in the Weight, Width and Italic data, to automatically generate this field using the **Build Style Name** button and edit this field if necessary
Build Style Name	Press this button to automatically generate the **Style Name** field. Style names are based on the Width, Weight and Italic information
Font Name	[name: 6] PostScript name. This name will be used by a PostScript print driver to reference the font. Do not include spaces in this name
Full Name	[name: 4] More detailed font name. It may include spaces as well as any other glyphs — this is the name that is exposed to users when the font is installed in Windows
Menu Name	The name used to access the font in applications. This name must not include style information (bold, italic or similar). The length of this field is limited to 27 characters for TrueType or single-master Type 1 fonts and to 7 characters for Multiple Master fonts. To ensure that the current Menu name is made properly, press the Check button
FOND Name	This name is used by the Mac OS to organize fonts into font families. Windows does not use it. We recommend you fill in this name if you plan to port your font to Mac by TypeTool for Mac or TransType
Build Names	Press this button to automatically generate the **Font Name** and **Full Name** fields. If you are creating a new font we recommend that you fill in the **Family Name** field, generate or manually fill in the **Style Name** field and press this button to create the Font and Full names. If necessary you can edit the names later.

Validate Names Press this button to check names.

Accessing MyFonts Database

The last button on the Basic names page is **Check at MyFonts.com**:
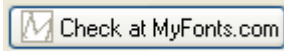
Click on this button to browse a huge database of fonts which are on sale at ***MyFonts*** (http://www.myfonts.com/), one of the world's largest online font distributors. This will allow you to check if your font name has already been used:
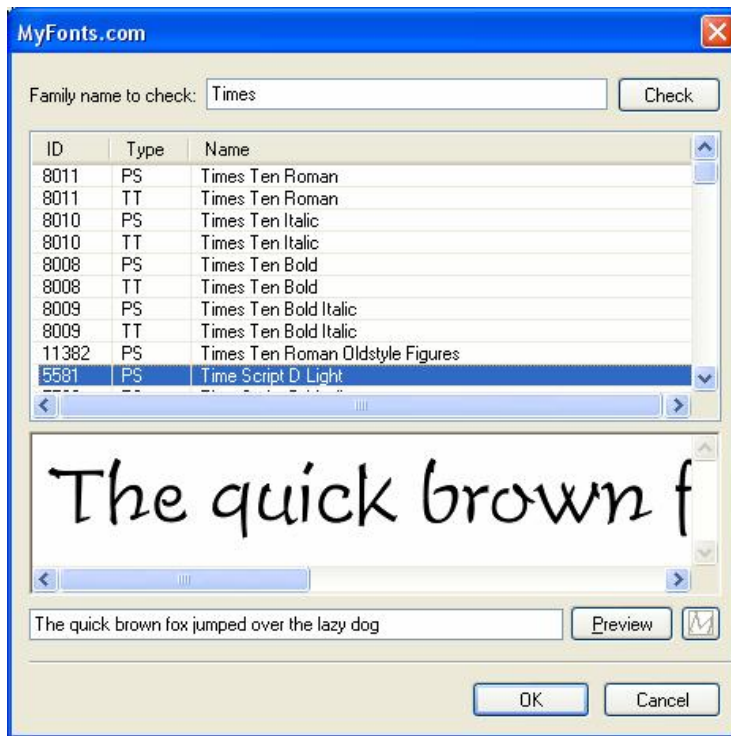
In the topmost editing field you may enter the font name that you want to check for similarities. Click on the **Check** button to send a request to the MyFonts database.

If you are connected to the Internet, MyFonts will give back with a list of font names (if any) that include the name you entered. Select one of the fonts and click on the **Preview** button (or double-click on the font name) to see a sample of the font in the sample box below the list.

The sample image is downloaded from MyFonts, so it may take some time (depending on the speed of your Internet connection).

You can modify the contents of the sample string in the editing field below the sample window. By default it has the standard "Quick brown fox..." sentence but you can enter anything there.

The button ▣ will open a browser window with the currently selected font presented in full detail:

There is also a **Buy Me** button — click on it if you want to buy the font.

How to Make a Font Family

Fonts on Windows may only include only 4 styles in a family. These styles are: Regular, Bold, Italic and Bold Italic.

If you have more than four styles in your Type 1 or TrueType typeface for Windows you must create several families. You may put all condensed styles into a Condensed or Narrow *sub-family* (like Arial Narrow, Arial Narrow Bold, Arial Narrow Italic), all black styles into a Black sub-family (Arial Black, Arial Black Italic) and all "normal" styles in the "Normal" sub-family (Arial, Arial Italic, Arial Bold and Arial Bold Italic).

Brief families on Windows with no more than 4 styles per family:
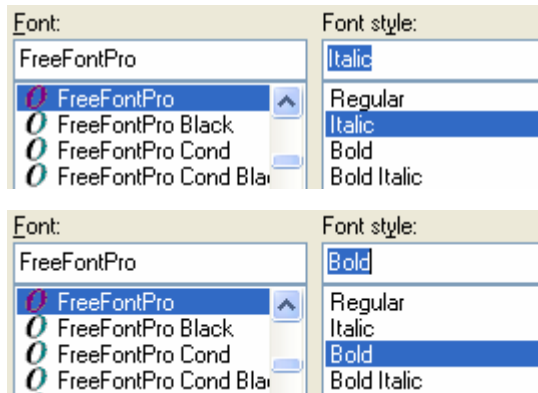
Mac Type 1 fonts and OpenType fonts (TT or PS) on Mac OS and within OpenType-savvy applications can contain an arbitrary number of styles within one family. To make OpenType fonts (TT or PS) cross-platform compatible, you have to create only families with up to 4 styles (as described above) in TypeTool.

In an OpenType font, the "brief" (Windows) family naming should be devised on the **Basic set of font names** page:

323

Copyright Information

On the **Font copyright** page you can enter information about the creators of the font. If you have created a new font you should enter your copyright notice here. If you have edited an existing font that was not your creation you must not remove the information contained on this page, or you may violate copyright laws.

Created by	Name of the company or person that created the font. If you are creating a new font enter your name or the name of your company here
Creation year	Year when the font was created. This is used by TypeTool to automatically fill in the **Copyright** field and is exported in TrueType fonts as the Creation year entry
Copyright	[name: 0] Copyright message. Must include the © sign or the word "Copyright", the name of the company or person that owns the copyright and the copyright year. In Type 1 fonts this information is stored in the Notice entry and in TrueType fonts in the Copyright entry

Trademark	[name: 7] Font trademark — used to save font's trademark notice
Build Copyright and Trademark Records	Press this button to create the standard Copyright record based on the **Created By** and **Creation Year** fields
Notice	[name: 10] Additional information that you want to include in Font Info. Exported in Type 1 fonts as the Copyright entry and in TrueType fonts as the Description entry.

Designer Information

This page stores information about the font's designer. Do not modify this data if you open an existing font to modify for personal use.

Designer	[name: 9] Name of font designer
Designer URL	[name: 12] A new entry implemented only in TrueType format. It is the WWW link to the designer of the font
Vendor URL	[name: 11] This TrueType-only entry shows the WWW link to the site of the font vendor.

Use the buttons to the right of the **Designer URL** and **Vendor URL** controls to open pages in a Web browser window. This requires an Internet connection.

License Information
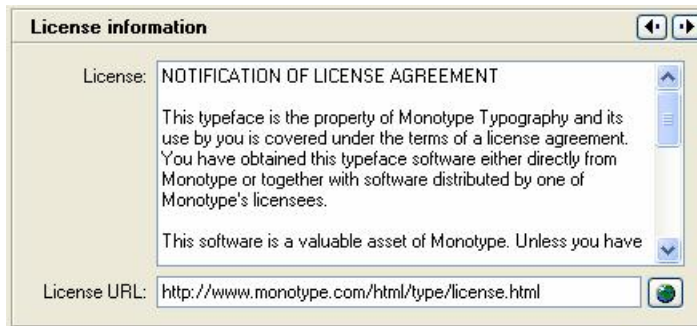
License and License URL records are relatively new and have appeared only in OpenType specification version 1.3.

License	[name: 13] License description — contains information about how the font can be used
License URL	[name: 14] URL where additional license information can be found.

Use the button to the right of the **License URL** control to open the page in a Web browser window. This requires an Internet connection.

Font Identification

Version and Identification
    Identification settings
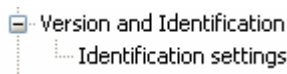
Sometimes the operating system or a DTP application needs to know what the font looks like. It may be necessary, for example, to properly substitute for a missing font with the closest look-alike.

TypeTool supports all the font-identification settings that are used in Type 1 or TrueType fonts.

Version Information

Font version and revision information	
Version: 2 Revision: 98	
Complete Version record: 002.098	
TrueType Version record:	
Version 2.98	

Version	Version of the font.
Revision	Revision of the font. Version and revision numbers are combined and build a complete version record that appears in Type 1 font headers
TrueType Version Record	[name: 5] TrueType font version records have a different format. You may enter the TrueType version record here or press the **Recalc** button at the right of the field to fill this record automatically. You must have the **Names** and **Copyright** pages filled in to use the automatic features on this page. Press the **Apply** button at the bottom of the dialog box to enter the new Font Info values into the font's header.

327

Basic Font Identification

TrueType Unique ID Record	This field is necessary to identify TrueType fonts. Usually it includes the creator's name, font family name and creation year. The format of this field is freeform, but we recommend that you use the 🢂 button to fill this field automatically
Type 1 Unique ID Record	An integer number identifying the font. Unique ID numbers must be registered with Adobe Systems. However, you may leave 0 in this field or enter a value from the users Unique ID zone (4000000 to 4999999). If you enter this value and plan to export Type 1 fonts, be sure not to have more than one font with the same Unique ID value because that may cause a problem with PostScript printers or Adobe Type Manager software
Type 1 XUID Numbers	More advanced identification codes for Type 1 fonts. This number is used only in PostScript Level 2 printers. Please, refer to Adobe documentation for more information concerning the **XUID** field
TrueType Vendor Code	An up-to-four letter length code that is assigned to most TrueType producers to identify their fonts. An *uppercase* vendor code must be registered with Microsoft or Apple. All registered Vendor codes known at the time of TypeTool's release are placed in the drop-down list box. If you want to identify yourself without registering you may enter a *lowercase* four-letter vendor code.
	Below the vendor selection list you can see the full name of the registered vendor. Click on the name to open the vendor's page in a Web browser
Use it as default	Check this option to use the current vendor code as the default in all new fonts. You may make your own code the default so you will not have to enter it every time.

Vendor.dat File

TypeTool stores information about registered vendors in the vendor.dat file located in the C:\Program Files\Common Files\FontLab\Data folder. This is a text file with a simple structure:

2REB 2Rebels
39BC Finley's Barcode Fonts
3ip Three Islands Press
918 RavenType

As you can see, it is   a vendor code followed by vendor name. A single space is used as a separator.

If you want to change the file or add a new entry, open it in any text editor (such as Notepad or WordPad) and make changes.

Metrics and Dimensions

```
Metrics and Dimensions
    Key dimensions
    TrueType-specific metrics
```
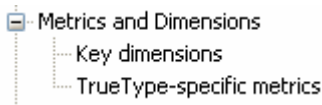
This page is used to set font dimensions that are used mostly to properly align text lines.

Font UPM Value

Font UPM value

Font's UPM size: 1000

☐ Scale all glyphs according to UPM size change
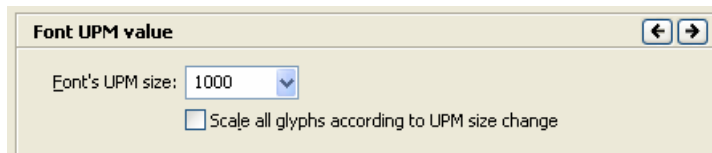
The most important field on this page is the **Font UPM size**. The UPM size defines the relation of the font to the point size in which the text is set, and defines the precision in which glyphs outlines are constructed. More details about the UPM size and some recommendations for its values are given in the section "*Units of Measurement* (on page 154)".

If you change the UPM value in the **Dimensions** page of the Font Info dialog box this does not necessarily mean that the size of the glyphs will change. After changing the UPM from 1000 to 2000 without scaling the glyphs, all the glyphs will be visually half as big as they were before. To keep the glyphs visually the same size, you need to scale them along with changing the UPM size — for that, enable switch on the **Scale all glyphs according to UPM size change** check box.

Basic Font Dimensions

The page has several editing fields with numbers and a sample window where an appropriate glyph is displayed to help set correct values.

Other fields on this page mean:

Ascender	Position of the font's ascender line. Usually this is the height of the lowercase 'b' glyph
Descender	Position of the font's descender line. Usually this is the position of the bottom line of the 'p' glyph
Caps height	Height of the font's uppercase glyphs. Usually the height of the 'H' glyph
x height	Height of the lowercase glyphs. Usually the height of the 'x' glyph
Italic angle	Actual italic or oblique angle for the font. The italic angle is measured in the counterclockwise direction, so the default value is -12?
Slant angle	Type 1 fonts can be artificially slanted to get an "oblique" appearance while keeping the actual outlines upright. Enter a slant angle value (in degrees) here and check the result in the Preview panel
Underline	This is the position of the middle of the underline line in your font
Thickness	This is the thickness of the underline line.

If you press the **Recalculate dimensions** button, TypeTool will automatically recalculate all the dimension values.

Ascender, Descender and Type 1 fonts

If you are making a Type 1 font you should set the Ascender and Descender values very carefully. In Type 1 fonts these values are used very directly to calculate interline spacing. It is usually necessary to set the Ascender value higher than actual height of the "ascender" 'b' glyph, to have some additional space between lines.

Advanced Vertical Metrics

In TrueType font files vertical metrics can be stored in the OS/2 and hhea tables. Different programs and operating systems use vertical metrics from these tables. Windows usually uses data stored in the OS/2 table while the Mac OS uses only data located in the hhea table.

It is important to correctly define all vertical metrics if you want your font to align properly. In most cases TypeTool can calculate vertical metrics according to the system recommendations, but in some cases you may want to customize these values.

We recommend you generally leave these values untouched in an existing OpenType font. Of course, if you perform heavy modification of the font you will need to update the advanced vertical metrics.

If you want TypeTool to automatically calculate all vertical metrics, select the option **Calculate values automatically**.

If you want to customize values, select the **Set Custom values** option and edit data in the fields below. Note that if you choose **Set Custom values** but leave all data unchanged, TypeTool will restore the original vertical metrics data from the imported font and the new updated font will align exactly as the original one.

Here is a description of each value:

Typo Ascender	This is the typographically-correct ascender value. It is the topmost line of lowercase glyphs, usually, the topmost line of the 'b' glyph
Typo Descender	The same as Typo Ascender, but for the lowest line. Usually it is equal to the bottom line of the glyph 'p'
Typo Line Gap	"Typographically" correct line gap value (distance between bottom line of the upper line of text and top line of the lower line of text)
WinAscent	[OS/2] This value defines the topmost line of all important glyphs in the font. "Important" glyphs are all non-exceptional glyphs. For example, if most of the glyphs have the topmost position at 900 font units and one, not often used glyph, has it at 1300 font units, it is a good idea to set WinAscent at 900 units. Note that in most cases portions of the glyphs that are above the WinAscent value will not appear on the screen or print on some printers. Please note that WinAscent is NOT a typography ascender, usually measured as the topmost line of lowercase glyphs. It is mostly a technical parameter used by the rasterizer to allocate vertical space to render glyphs
WinDescent	[OS/2] The same as WinAscent, but for the lowest line of all "normal" glyphs
Ascender	[hhea] This value is used by the Mac OS in about the same situation as Windows uses the WinAscent value from the OS/2 table — to define the topmost position of all important glyphs
Descender	[hhea] In short: the Macintosh version of the Windows WinDescent parameter. If there are any pixels below this line the glyph will be squashed in the vertical direction to match metrics defined by the Ascender and Descender parameters
Line Gap	[hhea] This value is used by the Mac OS to compensate Ascender and Descender values and calculate the correct distance between baselines of the text. Refer to the formulas below to see how baseline-to-baseline distance is calculated.

Baseline-to-baseline distance calculation

Windows:

Windows Metric	OpenType Metric
ascent	WinAscent
descent	WinDescent
internal leading	WinAscent + WinDescent — UPM
external leading	MAX(0, LineGap — ((WinAscent + WinDescent) — (Ascender — Descender)))

BTBD = ascent + descent + external leading

It should be clear that the "external leading" can never be less than zero. Pixels above the ascent or below the descent will be clipped from the glyph; this is true for all output devices.

Macintosh:

Macintosh Metric	OpenType Metric
ascender	Ascender
descender	Descender
leading	LineGap

BTBD = ascender + descender + leading

Encoding and Unicode

└─ Encoding and Unicode

As we mentioned earlier, fonts may have very many characters and support a lot of different languages. To tell the operating system what codepages the current font can support, you set the codepages information.

TrueType and Type 1 fonts use different methods to identify what codepages a font supports. In TrueType fonts you can identify all the supported codepages by setting bits in a special field of the font header. In Type 1 fonts you select only one codepage (actually, encoding vector) and it must be compatible with the actual font encoding.
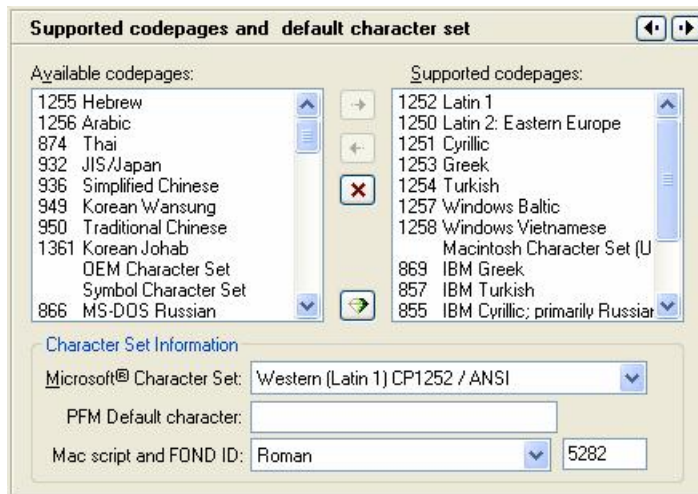
To select the supported codepages automatically press the ⬇ **Auto** button. TypeTool will analyse the Unicode information available in the font and will automatically detect which codepages this font can support.

To add a codepage to the list of supported codepages select a codepage in the left list and press the → **Add** button.

To remove a codepage from the list of supported codepages select a codepage in the right list and press the ← **Del** button.

To reset the list of supported codepages, press the ✖ **Reset** button.

The Meaning of Supported Codepages in Windows

In Windows 3.1x this information is not used.

In Windows 95 and Windows NT a font that has more than one standard (1252 Latin 1) codepage supported will appear as a font available for different scripts. So, if, for example, you set Latin 1 and Cyrillic codepages for a font with the name MyFont, in Windows 95 (and NT) it will appear as MyFont (Western) and MyFont (Cyrillic).

Type 1 Character Set

Type 1 fonts do not have such extensive support for multiple codepages. The glyph names they use to identify glyphs are mapped to codes through the encoding vector. There is one parameter that is used to tell Adobe Type Manager (used to support Type 1 fonts in Windows) how to interpret the encoding vector. This is the Microsoft Character set.

Some of the values for Microsoft Character set:

ANSI	The Font has all the glyphs necessary to represent the standard Windows Latin 1 character set. No reencoding is necessary
Symbol	The font is symbolic (dingbats) with a custom encoding vector. It should appear as Symbol in Windows applications and the font's own encoding vector should be used to access glyphs
ShiftJIS	This is a Japanese font that includes Kanji glyphs
OEM	The font has MS DOS glyphs. This setting is rarely used in Type 1 fonts
Bitstream	This is a normal text font, but it has its own encoding that should be used to access glyphs. This setting is highly recommended for all text fonts with a non-standard encoding vector
Arabic	The font has Arabic encoding.

Other values cover more codepages that may be supported by the font. Choose the codepage that is the default for your font.

Printing and Proofing Fonts

TypeTool 3.0 has several new tools for visual proofing and printing of your fonts. Several new printing modes allow you to print font content in different ways with various options. You can print all or selected glyphs with all character information, sample strings and detailed glyph printouts. A new Quick Test feature that proofs the font using the system renderer has also been added.

Printing

To see the printing modes available, choose **File > Print**. You will see the following dialog box:
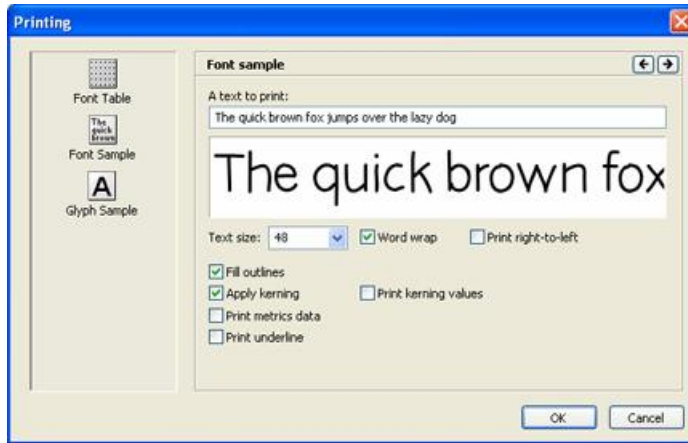
On the left side of the dialog box, you can choose one of the available printing modes: Font Table, Font Sample and Glyph Sample.

Printing Font Table

To print the font table of the current font:

1. Select the **Print** command in the **File** menu or click on the 🖶 button in the Standard toolbar. You will see a dialog box that asks you to select one of the printing modes:

2. Select the **Font Table** item in the list at the left to print a font chart containing samples of all font glyphs.

3. Choose how many cells you want to be printed in one row. The fewer cells will be printed in a row the larger will be each cell and the more pages you will get printed.

4. Choose information for the cells caption. This option is similar to the caption context menu in the Font window.

5. Select whether to print glyph index in every cell or not and click on **OK**. You will see the standard Windows Print dialog box that will ask you to choose a printer and modify the printer settings:
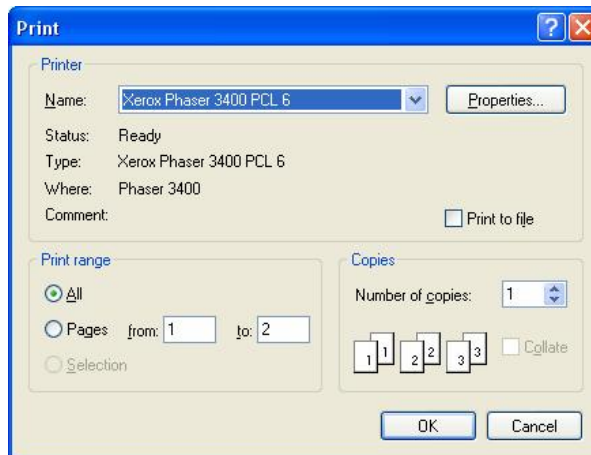
In this dialog you can choose the range of pages you want to print.

341

When you click on the **OK** button, TypeTool will print a font table containing samples for all font glyphs.

Printout of the font chart:

Printing Font Sample

To print the current font sample:

1.   Select the **Print** command in the **File** menu or click on the 🖨 button in the Standard toolbar and select the **Font Sample** item at the left of the Printing dialog box:

2.   At the right type the text you want to be printed and set the size of text. Note that metrics information is printed only when larger sizes are selected.

3.   Set other printing options:

Word wrap	If this option is on TypeTool will wrap words when printing
Fill outlines	If this option is on characters will be printed filled
Apply kerning	If this option is on and the kerning is defined for the font the text will be printed kerned
Print kerning values	If this option is on and the kerning is defined TypeTool will print kerning values
Print metrics data	If this option is on and the text size is large enough the values of the glyph advance width and sidebearings will be printed
Print underline	If this option is on the text will be printed underlined.

4.   Click on **OK**. You will see the standard Windows Print dialog box.

Printout of the font sample with default printing options:

Printing Glyph Sample

To print the glyph sample:

1.   Select the **Print** command in the **File** menu or click on the 🖨 button
in the Standard toolbar and select the **Glyph Sample** item at the left of
the Printing dialog box:
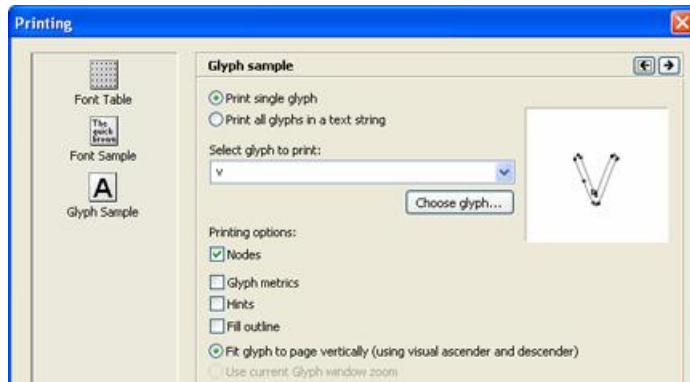
2.   At the right select the single glyph that you want to be printed or type
the names of glyphs proceeded by slash. You also may use glyph
Unicode codepoints proceeded by backslash. You can use the **Choose
Glyph** button to open the Find Glyph dialog box.

3.   Set other printing options:

Nodes	If this option is on TypeTool will print squares visually representing nodes
Glyph metrics	If this option is on TypeTool will print lines representing glyph metrics
Hints	If this option is on and hints are present they will be printed
Fill outline	If this option is on glyph outlines will be printed filled
Fit glyph to page vertically	If this option is selected each glyph will be printed scaled to fit the page vertically and centered horizontally
Use current Glyph window zoom	If this option is on each glyph will be printed at the size of the currently opened Glyph window. This option cannot be selected if there are no opened Glyph windows.

4. Click on **OK**. You will see the standard Windows Print dialog box.

Use combinations of different options to see the printing result which better meets your needs.

Printout of the glyph sample with default printing options:

TYPETOOL GLYPH SAMPLE   Monotype Typography, Inc.

Font: Arial-Black   09/14/06 12:31:49

Glyph: A   Page 1/1
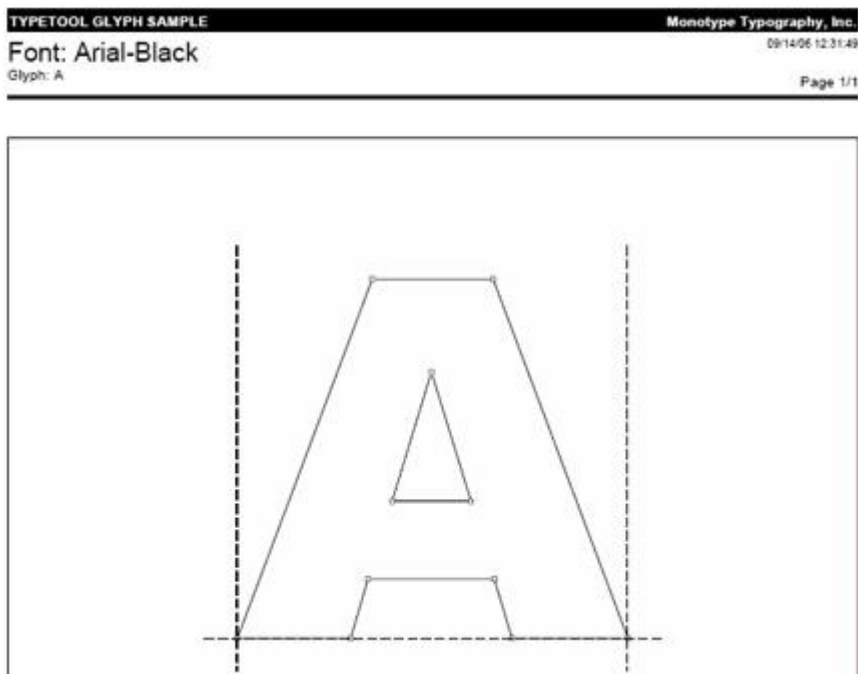
Quick Test

TypeTool has a new feature allowing you to test your fonts quickly. You can generate and temporarily install your font with one command and then see how it works in real text-editing environment.

To test the open font, select the **OpenType TT** or **OpenType PS** command in the **Tools > Quick Test As** menu. TypeTool will generate the font in the selected font format (using the current generation Options), temporarily install it in your system, and present the following window:

This is a simple text editor allowing you to type with your font, change the text size and print the content.

Type or paste the sample text in the editing field. Use the **Content** context menu to see the characters of the particular codepage or entire character set of the font, i.e. all glyphs that have a Unicode codepoint assigned.
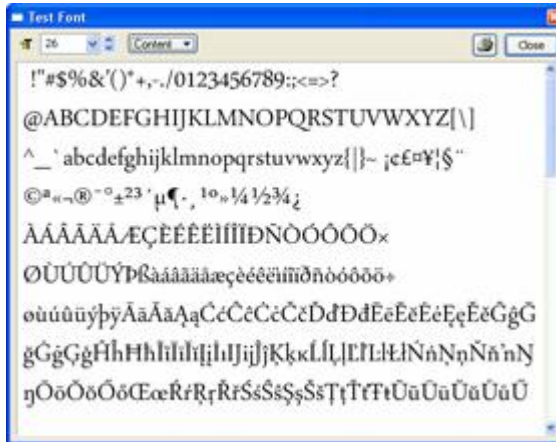
You can also type in your custom text. Note that since this dialog uses the system font rendering, you will see the OpenType layout features that are applied by the operating system by default.
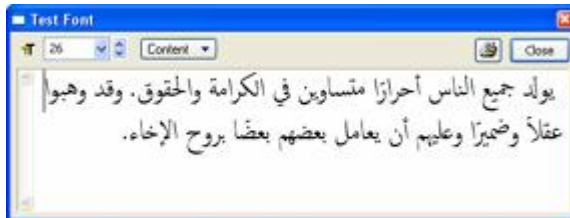
You can print the display text from this dialog.

You can launch other applications (e.g. MS Word or InDesign) and see how the font works there. The font name is generated automatically and starts with 'FLSFNT'.

To finish testing the font and close the window, click on the **Close** button.

Generating Fonts

In this chapter, we will discuss the most important aspects of generating workable fonts in the most popular formats. Please note that all recommendations and guidelines in this chapter only address typical and common cases. There can be exceptions and special situations. For those, you need to refer to the specific sections of the manual, and to the appropriate font format specifications.

This chapter also assumes that you have read the rest of the manual.

Relevant Font Formats

The following lists the most relevant font formats and lists some of their advantages and disadvantages.

OpenType PS

Also known as: OpenType-CFF, PostScript-flavored OpenType, OTF

Filename extension: .otf

Pros: Works on Windows, Linux, Mac OS 8.6, 9, and OS X. Uses the Bézier curves that are preferred by designers and used in drawing apps such as Illustrator and Freehand so letterforms can be drawn precisely and outlines need not be converted. May contain up to 65,535 glyphs, supports Unicode and can contain OpenType Layout features.

Suitable for Western Roman fonts, non-Latin fonts, multilingual fonts and advanced typography. May include class kerning allowing for moderately-sized kerning tables. Uses Type 1 hinting that is relatively easy to create. Can include embedding rights information defining whether or not the font may be attached to electronic documents.

Cons: Type 1 hinting does not allow precise control in small screen sizes. Can theoretically contain bitmaps, but they are not displayed. Since this is a relatively new format, there are problems with old some applications (some styles are not displayed in menus, kerning for non-Western characters does not work.) The multilingual and advanced typography features only work with new OpenType-savvy applications, otherwise just the basic character set is available. Two alternative family namings within each font must be devised: one where a family contains an arbitrary number of styles, and second "brief family" where one family does not contain more than four styles.

Windows TrueType / OpenType TT

Also known as: data-fork TrueType, Windows TrueType, TrueType-flavored OpenType, TTF

File extension: .ttf, also possible: .otf

Pros: Works on Windows, Linux and Mac OS X. May contain up to 65,535 glyphs, supports Unicode and can contain OpenType Layout features.

Suitable for Western Roman fonts, non-Latin fonts, multilingual fonts and advanced typography. May include class kerning allowing for moderately-sized kerning tables. TrueType hinting allows precise control in small screen sizes, can also contain bitmaps. Can include embedding rights information defining whether or not the font may be attached to electronic documents.

Cons: Does not work on Mac OS 8/9. May cause output problems on ten-year-old PostScript output and printing devices. The designer usually needs to convert the outlines from Bézier curves which may introduce very slight changes in the shape. When converted back to Bézier curves (e.g. in Illustrator), the resulting curves have superfluous points. Manual TrueType hinting is laborious to create. The multilingual and advanced typography features only work with new OpenType-savvy applications, otherwise just the basic character set is available. For font families, requires two versions of the family name within each font: the first may contain any number of styles; the second "brief family" may contain only four styles.

351

Windows Type 1

Also known as: Windows PostScript, PC PostScript, PC Type 1

File extension: .pfb, with supplementary files .afm, .inf, .pfm

Pros: Works on Windows and Linux. Works in all PostScript commercial output and printing devices. Uses the same curve system (Bézier) as drawing applications such as Illustrator and Freehand, so letterforms are easy to edit when converted to curves. Type 1 hinting is comparatively easy to create.

Cons: Does not work on Mac OS 9 or X, not cross-platform. Contains two parts, the outline file (.pfb) and the metrics font (.pfm), both of which must be in the same folder. Does not contain class kerning so kerning tables are large. Type 1 hinting does not allow precise control for very small screen sizes. Cannot include more than 256 encoded characters and lacks advanced layout features such as ligatures, making the format unsuitable for multilingual or non-Latin fonts. Cannot contain bitmaps for small screen sizes. Does not contain embedding rights information. One family cannot contain more than four styles.

Recommendation: Draw fonts with Béziers as Type 1. When complete, make a duplicate TypeTool master .vfb file and make TT conversions to it. Generate either a TrueType / OpenType TT or an OpenType PS font for the newest systems (Windows and Mac OS X).

Mac Type 1 is not included in this overview since TypeTool for Windows cannot generate Mac Type 1 fonts. You can use TypeTool for Mac or TransType for Mac or Windows to produce a Mac Type 1 font. However, we recommend producing fonts in the OpenType format unless you have Mac customers running a pre-8.6 Mac OS.

Before You Generate

Before you generate your font, make sure the most relevant aspects of the font are complete. Open all fonts that belong to your family in TypeTool.

Font Info

Open **File > Font Info** for each font. On the **Names and Copyright** page, make sure that all text boxes and combo boxes are filled in and that the **Font is bold** and **Font is italic** check boxes are checked accordingly. Use the **Validate Names** button for each font to check against potentially wrong names.

Check the pages **Copyright information, Designer information** and **License information**. All information should be filled in there (although for Type 1 fonts, actually only a subset of the entries will be included in the generated fonts).

In **Version and Identification**, put a reasonable version number into the top fields and click on the green **Auto** button. Increase your minor version number if you revised your font. On the **Basic identification** settings, click on the green **Auto** button and on the **Now** button. Leave the Type 1 number fields empty, select your vendor ID from the **TrueType vendor code** combo box. If you do not have one, register one at Microsoft Typography or leave the default value.

On the **Metrics and Dimensions** page, font UPM size can be 1000 for all formats. do not worry about things like 2048 for TrueType.

On the **TrueType-specific metrics** page, click on both **Recalculate** buttons. Then go to **Key dimensions** and check **Copy values to TrueType metrics**. Make sure your **Ascender** and **Descender** values are uniform for all fonts in your family; use average values if your styles de facto have different ascenders or descenders. The Ascender value should be positive, the Descender value should be negative (preceded with a minus sign) in the fields. Make sure the sum of the absolute values of Ascender and Descender are equal to the font UPM size, e.g. if your Ascender is 720 and your UPM size is 1000, your Descender should be -280.

Go again to the **TrueType-specific metrics** page and make sure the values of **OS/2 WinAscent** and **OS/2 WinDescent** are uniform across your family, average if necessary. The value of all **OS/2 Typo** values should be uniform as well. **TypoLineGap** should be between 5% and 20% of your UPM size and uniform across all styles.

The **hhea Ascender** and **hhea Descender** should have the values equal to **WinAscent** and **WinDescent**, and the **hhea LineGap** should be 0. Alternatively, the **hhea Ascender**, **hhea Descender** and **hhea LineGap** should be equal to the corresponding Typo fields.

You can usually leave the other Font Info settings that were not mentioned above at their factory settings.

Character Set

Switch to the **Names mode** in Font Window and select some encodings to see whether all glyphs in a desired encoding that you wish to cover by your font are included in the font. If you are making a text font, it should at least cover all glyphs from the MacOS Roman and the MS Windows 1252 Western (ANSI) encodings.

You may want to create a custom .enc Encoding file that will work as your font family "map" and will include all glyphs that you want to include in the font.

Switch to the **Codepages mode** and check several codepages. If you are making a text font, it should at least cover all glyphs from the MacOS Roman and the MS Windows 1252 Western (ANSI) codepages.

Remember that the Names mode uses glyph names and the Codepages mode uses Unicode codepoints to reference the glyphs. Your font should have both the glyph names and the Unicode codepoints conform to published recommendations.

If any glyphs are present "in the yellow zone" in the Codepages mode but are missing from the "yellow zone" of the corresponding Encoding, your glyph names may be incorrect. Choose **Glyph > Rename Glyph** to fix this problem.

If any glyphs are present "in the yellow zone" in the Encoding mode but are missing from the "yellow zone" of the corresponding Codepage, your Unicode codepoints may be incorrect. Choose **Glyph > Generate Unicode** to fix this problem.

It is recommended that all fonts in your family have the same character set.

In Font Info, on the **Encoding and Unicode** page, click on the green **Auto** button. If you are making non-Western single-codepage Type 1 fonts or a Symbol-encoded TrueType font, select the appropriate character sets from **Microsoft Character Set** and **Mac script and FOND ID** combo boxes.

Glyphs

Select all glyphs in the Font Window (**Edit > Select All**). Choose **Contour > Correct Connections**.

If you are making an OpenType PS or Type 1 font, choose **Contour > Convert > Curves to PostScript** (if it is enabled). Choose **Contour > Paths > Set PS Direction**.

If you are making an OpenType TT, choose **Contour > Convert > Curves to TrueType** (if it is enabled). Choose **Contour > Paths > Set TT Direction**.

Hints

If you are making an OpenType PS or Type 1 font, choose **Tools > Hints & Guides > Autohinting**.

You can now review your hinting manually or leave it as is — TypeTool will take care of the rest as good as it can.

Kerning

Open the **Metrics Window** and review your kerning.

Options for Converting Fonts

We recommend selecting different options for converting fonts between different formats. In addition to the font generating options, we also suggest particular opening options that will produce the best results in specific situations:

Source	Destination	Opening options	Generating options
TrueType / OT TT	TrueType / OT TT	Do not convert curves, do not scale to 1000, do not decompose, do not autohint, store custom tables, store native hinting	All hinting options — on, write custom tables
TrueType / OT TT	Type 1	Convert curves, scale to 1000, do autohint, do not decompose	Write PFM, AFM and INF files, Select encoding automatically. Before export, switch the Font window to the Names mode and select the desired encoding vector
TrueType / OT TT	OT PS	Convert curves, scale to 1000, do autohint, do not decompose, store custom tables	Autohinting on, Decompose on
OT PS	Type 1	Do not decompose	Write PFM, AFM and INF files, Select encoding automatically, Autohinting off
OT PS	OT PS	Do not decompose, store custom, do not scale to 1000	Autohinting off, Decompose off, write custom tables
OT PS	TrueType / OT TT	Do not decompose, read all records, read OT, store binary OT, store custom, do not interpret	All hinting options — on, do not reencode first 256 glyphs
Type 1	Type 1	Do not decompose, Generate Unicode	Write PFM, AFM and INF files, Select encoding automatically
Type 1	OT PS	Do not decompose, Generate Unicode	Autohinting off, Decompose on
Type 1	TrueType / OT TT	Do not decompose, Generate Unicode	All hinting options — on

Of course you can choose other options, but when you just want to convert a font from one format to another these recommended combinations of opening and generating options will usually give you fonts that will work fine in most environments.

OpenType Fonts

In this chapter we will discuss working with the OpenType fonts. The OpenType font format, jointly developed by Microsoft and Adobe, allows us to combine the best features of the TrueType and Type 1 font formats.

OpenType fonts are stored in a single font file, use Unicode as their encoding and work in Windows and Mac OS X.

This all has been true for older TrueType fonts but the advantage of OpenType against older font formats is the support of layout features, which allow better typographic layout, and precise support of complex scripts.

Font Features

OpenType fonts come in two formats, sometimes called flavors, OpenType TT and OpenType PS. Both sorts of OpenType fonts may include so-called *OpenType Layout features*. The layout features are rules that change the standard behavior of the font.

For example, the small caps layout feature (abbreviated *smcp*) may change all lowercase glyphs to their small caps counterparts.

Effluent
EFFLUENT

Small caps

The standard ligatures layout feature (abbreviated *liga*) can replace some letter combinations with ligatures.

Effluent
Effluent

Ligature

The old-style numerals layout feature (abbreviated *onum*) can replace lining figures with old-style figures.

12345
12345

Old Style Numerals

OpenType Layout features can serve typographic purposes like shown above. In this case, applications such as Adobe InDesign, Adobe Illustrator CS, Adobe Photoshop CS, Apple Pages or Apple Keynote on Mac OS X 10.4 offer the user some user interface to turn selected features on an off.

OpenType Layout features also play a crucial role in rendering complex scripts, i.e. writing systems such as Arabic, Devanagari or Thai. These writing systems have complex rules for displaying characters. For example Arabic uses different forms of letters if a letter is found at the beginning, in the middle or at the end of the word. Also, complex scripts often use vowel marks that are positioned dynamically over consonant letters. In all these cases, the layout features contain mapping rules that are automatically applied by the layout application.

Note that not all layout applications offer the same level of OpenType support. For example, Microsoft Word 2003 for Windows supports complex-script layout features for Arabic and Devanagari but does not support Western typographic layout features. Adobe InDesign CS2 U.S. English and Apple Keynote on Mac OS X support Western typographic layout features but do not support any complex-script layout features. Adobe InDesign CS Middle East edition supports Western and Arabic layout features, but does not support Devanagari.

Information about using OpenType fonts can be found at:

http://www.myfonts.com/info/opentype/
http://store.adobe.com/type/opentype/

Information about developing OpenType fonts can be found at:

http://www.microsoft.com/typography/SpecificationsOverview.mspx
http://www.microsoft.com/typography/developers/opentype/
http://partners.adobe.com/public/developer/opentype/

Probably the best thing about OT features is that they do not change the source string of characters. To explain this we need to again talk about the character-glyph model.

The source text that you type on the keyboard or get from another source is a sequence of characters that have strong links to the codes that the computer uses to store data. The image of the text that you see on the screen is a sequence of glyph images. It is important to understand that there is not necessarily a one-to-one relationship between character and glyph: it is possible to have a single glyph used as the image for more than one character (Latin A and Cyrillic A, for instance, are different characters, but use the same glyph) and sometimes you may have more than one glyph "serving" a single character.

Please, remember this key OpenType principle: the OpenType Layout engine does not know anything about characters! All the features that OpenType can have are defined for glyphs. This is the process of OpenType text processing:

0. As a source we have a sequence of characters.

1. Character codes are mapped to default glyphs using the Unicode mapping table. In TypeTool this is what you see in the Font window when you select one of the codepages. Here we have a sequence of characters replaced by a sequence of glyphs. No character information is available beyond this point!

2. The source sequence of glyphs is passed to the OpenType processing module, which then applies the font features in a pre-defined sequence. The list of the features to apply is determined by the application (for example, in Adobe InDesign you can explicitly select features to apply) or operating system (e.g. the rendering of Arabic text with an OpenType font).

3. The resulting sequence of glyphs is passed to the second stage of feature processing which can shift the positions of glyphs. Kerning is applied at this stage.

4. The sequence of glyphs, accompanied by the positioning information, is passed to the rasterizer, which does the imaging of the features on the destination device: screen or printer.

OpenType Font Formats

Another key feature of the OpenType format is the fact that from the user's point of view there is only one font format for Mac, PC or any other platform.

From the inside, there are two possible forms of OpenType fonts: **OpenType TT** and **OpenType PS**.

The general structure of the font file is the same and both versions of the format provide the same functionality. There are some technical differences:

Version	OpenType TT	OpenType PS
Outlines	2nd-order, like in TrueType fonts	3rd-order, like in Type 1 fonts
Hinting	TrueType instructions	Type 1 declarative hints
File extension	.ttf (but may be also .otf)	.otf
Comments	Technically, these fonts are an extension of the PC TrueType format and are backwards compatible with them. Therefore, in TypeTool we refer to them as TrueType / OpenType TT. From the practical point of view, any PC TrueType font is automatically an OpenType TT font and vice versa.	Outline data is stored in a CFF (Compact Font Format) table. When printed to a PostScript device, the font is converted to Type 1 so it is backwards-compatible with all PostScript devices.

363

What Format to Prefer

It is not easy to say which version is better. Both formats will work on both platforms. For Windows-centered office use we would recommend TT-flavored OpenType fonts, as they will provide better compatibility with the old versions of the OS.

For cross-platform and DTP-oriented applications OpenType PS fonts seem to have some advantage because they will provide better outline quality in Bézier drawing (less outline points). On the other hand, OpenType TT fonts may theoretically be delta-hinted and therefore have excellent screen quality.

Please note that the differences are minor and the most important thing to choose is the source format in which you have your fonts. If you have Type 1 fonts, it will be easier to convert them to OpenType PS.

OpenType and TypeTool

Support of OpenType fonts in TypeTool is quite limited. You cannot edit or create OpenType Layout tables in TypeTool.

OpenType support may be separated into two stages:

1. Importing OpenType fonts and reading the binary OpenType tables. TypeTool may store the original binary tables in the .vfb file.

2. OpenType fonts export. At this stage the Adobe FDK for OpenType (AFDKO) library is used to build the OpenType font files. If possible original binary OpenType tables are exported.

To edit imported or create and compile new OpenType Layout tables you will need *FontLab Studio 5* (http://www.fontlab.com/studio/) that is the "bigger brother" of TypeTool.

Importing OpenType Fonts

There is nothing special about importing OpenType fonts: use the **File >
Open** command to open files with .ttf extension (OpenType TT fonts) or
.otf extension (OpenType PS fonts).

When reading OpenType fonts TypeTool stores original binary OpenType
Layout tables — they will be stored in your .vfb file. This is useful if you
wish to make some changes to an existing OpenType font (e.g. add or fix
the design of some glyphs) but you do not want to touch the OpenType
Layout tables. Remember that TypeTool cannot decompile OpenType
tables.

Disable the following option to prevent TypeTool from storing the
additional custom TrueType/OpenType tables in the .vfb file:

☑ Store custom TrueType/OpenType tables

Generating OpenType Fonts

Before saving an OpenType font file you need to check the OpenType generation options in the **Tools > Options > Generating OpenType & TrueType** dialog box:

Generating OpenType PS, OpenType TT and TrueType	← →
☐ Automatically reorder glyphs	
☑ Write stored custom TrueType/OpenType tables	

Automatically reorder glyphs	If this option is enabled, TypeTool will try to reorder glyphs to match the Mac cmap encoding table. Technically, this is a requirement of the Apple TrueType specification but it is not required on Mac OS X or Windows
Write stored custom TrueType/OpenType tables	When enabled, stored custom TrueType/OpenType tables will be written into the generated font.

These settings only apply to fonts that you generate in the OpenType PS (.otf) format:

Generating OpenType PS (.otf)	← →
☑ Decompose all composites	
☑ Use subroutines to compress outlines in the CFF table	
☑ Autohint unhinted glyphs	

Decompose all composites	When enabled, all composite glyphs in the font will be decomposed. Recommended for maximum compatibility.
	When disabled, the composite glyphs will be exported as such
Use subroutines to compress outlines in the CFF table	Allow to automatically generate outline subroutines if font is generated as CFF-flavored. Outline subroutines store repetitive parts of outlines and allow to reuse with references from outline definition code
Autohint unhinted glyphs	When enabled, all glyphs that contain no hints will be autohinted.

After all options are set correctly, use the **File > Generate Font** command to save the OpenType font file. Select **TrueType/OpenType TT (*.ttf)** to generate an OpenType TT (TrueType-flavored) font or **OpenType PS (*.otf)** to generate an OpenType PS (CFF-flavored) font.

Index