Printed in the United States of America.

preface

Welcome to Textures®

This guide describes Textures®, a programmable publishing system for the Macintosh computer. Textures is a complete, interactive Macintosh implementation of Donald Knuth's TEX typesetting language, with a host of added features.

Installing Textures

Textures is easily installed with the Installer program on your "Textures CD" disk. For specific installation instructions as well as details about the distribution files that came with your Textures package, see the companion *Textures Installation Guide*.

About this Book

Once you have installed Textures, this is the book to turn to for a complete description of the Textures system. For information on TEX itself, see *The TEXbook* by Donald Knuth and the other books listed on page 224.

If you're not already familiar with basic Macintosh operations and terminology, it would be helpful to review your *Macintosh User's Guide* before beginning to use Textures.

Getting Started with Textures For a quick start, you can turn to Chapter 1, "Basics," which depicts the typesetting process in broad strokes (page 35).

Users who are new to TEX will benefit by reading on through Chapter 2, "Expanded Basics." New users can then proceed to Chapters 3 and 4, "Editing" and "The Macros Menu," and finally, read the first two sections of Chapter 5, "Formats," for a basic start with Textures.

Experienced TEX users may want to skim through Chapters 1 and 2, for an overview of the Textures interface, and then read Chapter 4 for an explanation of the Macros menu.

The topics covered in the remaining chapters can be read in any order.

Some Conventions

In this manual, `monospaced` type indicates a TEX command that would appear literally as part of a TEX file. For example, `\baselineskip=12pt` is an example of a TEX command that sets the spacing between lines. `Monospaced-italic` type is used to indicate a variable argument to a command. Brackets (`[ ]`) indicate optional arguments to a command. All other characters such as “ , `{` , `}` , and so on, are to be entered literally as you see them. For example,

```
\special{illustration picture-name [scaled nnnn]}
```

describes the form of a command used to include encapsulated PostScript graphics in your file. This might yield the actual command:

```
\special{illustration Figure001 scaled 750}
```

Finally, **boldface** type signifies a special term that is defined in the Glossary beginning on page 226.

table of contents

Abbreviated

table of contents

Detailed

update

What’s New in Textures 2.0

This chapter describes the new features of version 2.0 of Textures. Here you will find general information and detailed specifics of the changes in Textures 2.0. This chapter supercedes information in the following chapters. (We will integrate the information in a future version of the User Guide.)

Remember Textures 2.0 requires a Power Macintosh to take advantage of all of its features. In particular, Synchronicity, Follow Focus, and Command-Click do not work on 68K machines. Also, since formats created on 68K machines cause Textures to use the 68K engine even when running on Power Macintosh machines, you’ll need to update old formats to PowerPC.

We are delighted with the new capabilities of Textures 2.0. Writing and revising TEX documents has never been so fluid, so easy.

Textures 2.0 is packed with new capabilities designed to help you create and edit TeX documents more quickly, easily and more efficiently than ever before. Effortless editing and navigation between different parts of your document saves you time. Clicking back and forth between TeX code and typeset result shows you which lines of code create which output, and vice-versa. Textures 2.0 gives a big boost to your readers by adding hyperlinking within documents and to outside URLs. Automatic font recognition lets you use installed fonts without creating metrics. and lets you add font style modifiers on the fly.

Options Menu

The typeset menu has changed with Textures 2.0. The Flash Mode menu item has been replaced with an **Options menu**. The Options submenu allows you to enable or disable Flash Mode, Synchronicity, Follow Focus and LaTeX mode for each document. Flash Mode no longer automatically typesets a document upon opening, but waits for a change to be made to initiate typesetting. Synchronicity, Follow Focus and LaTeX mode are described in detail below. Flash Mode, Synchronize, and Follow Focus are automatically enabled in Textures 2.0; to change the default settings for new documents, open and modify the `Defaults` document located in the `TeX Formats` folder.

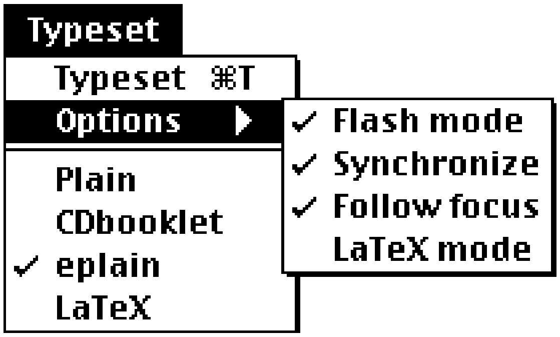


Figure 2-1. Textures new Options Menu.

Synchronicity

Synchronicity is the trademark technology of Textures 2.0, accumulating information used to cross-link or “synchronize” the edit window(s) of each document to its typeset output. The synchronizing data is collected as the document is typeset, enabling Command-Click to move between the Edit and Typeset

Windows, and Follow Focus to keep the Typeset Display synchronized with editing changes.

Remember, Synchronicity requires a PowerPC processor. It won't work on 68K machines, or with formats that were compiled on 68K processors.

Command-Click (`Click`) With the command key down, a single mouse click in an edit window moves the typeset display to show the corresponding position — and vice-versa, command/click in a typeset window moves to the corresponding point in the edit window, auto-opening the file if necessary. (It's easier to play with this than it is to explain it!). If the document has not been typeset, or has been changed since typesetting, the crosslinking information will not exist, or will be out of date. In that case, an alert box instructing you to `Please typeset the document` will appear. Simply typeset the document and try command-clicking again.

Note that, if the corresponding position is already visible on-screen, no actual movement may be needed — but it's working. Textures readjusts the page position only to make new data visible - otherwise the Typeset Window would appear "jumpy," constantly readjusting with the Follow Focus feature. It'll beep if it can't find a corresponding location, and sometimes it may present an unexpected correspondence; TEX macros can do strange things!

Note: The hand icon that scrolls the typeset window has been moved. It now is connected to the key combination of the command and option keys together.

Follow Focus

Upon completion of typesetting,the typeset window automatically displays the page with the most recent changes. Follow Focus keeps the Edit window and the Typeset window in sync, by automatically positioning the Typeset Window to the active point of the Edit window. In Flash Mode, it runs continously, tracking changes as they are made.

LATEX mode

LaTeX Mode is a new addition to Textures 2.0, When selected in the Options Menu, a typeset command will cause Textures to run two complete, consecutive typesetting runs on the document. This functionality is very useful for LATEX documents and other files with indexing and complex referencing.

Project Master

Sometimes a document will include a variety of input files, all of which are called upon by TEX to create a single typeset file. The first time the master document is typeset, Textures creates a list of the included sub-files and keeps them in memory. These files together are considered a project and the main file is the project master. After a document has been typeset once, and so long as the master (top-level) file is open, Textures remembers the entire document structure with all of its input files. When you tell Textures to typeset an input file, or make changes to a file in Flash Mode, it will typeset the complete document. Conversely, if you command-click in the typeset display of a project document, Textures will automatically open the input file and display the source code.

Automatic Font Installation

Textures 2.0 allows you to use any installed font without creating or installing metrics. Textures automatically creates font metrics for installed Macintosh fonts (those available to all Macintosh applications). The “Show Fonts” window will display their names (the same as other applications), and allow you to command/copy a font name to paste into the Edit Window. Since TEX doesn’t allow spaces in font names, space characters may be represented by the ellipsis (`É`) character, option/semicolon. So, for example, if your Macintosh had the font `Times New Roman` in its installed set, it would be entered in a font command as `TimesÉNewÉRoman`.

The font encoding for Macintosh fonts is typically the Macintosh standard, although it is font-specific; it cannot be changed. Since most mathematical and symbol fonts used with TEX use TEX encoding, separate metrics will still be required to insure correct character assignment for these specialty fonts.

Modifying Font Styles

Font styles may now be added directly into font definitions within your documents. For example,

```
\font\sample=times[bold,italic] at 8pt
```

will create ***a new bold, italic face*** for use within your document. Style modifiers are included as part of the font name, enclosed in brackets; additional styles are separated by commas, and may appear in any order. Recognized style names are ‘bold’, ‘italic’, and ‘outline’ There is also a ‘track+000’ style modifier that adjusts the inter-letter spacing by an amount specified in thousandths of an em-space. For example, `\font\wide=times[track+500] at 9pt` creates a font that has wide letter spacing. For more information on font style modifiers, see the Sample Font Modifiers document on the CD.

Edmetrics Update

Existing font metrics can now be directly accessed through Show Fonts, by double-clicking on a selected font name in the Show Fonts dialog box. The font metric for that font will open, and may be modified. Changing the TeX Metric Name will create a new font metric. NOTE: If you change features of an existing font metric, and do not change the name of the metric, you will be overwriting the existing metric! It’s a good practice to rename any metrics that you alter. Once you’ve made the desired changes, click OK and the new or revised metric will be added to Textures font list. Because metrics are loaded into Textures at startup, metric modifications will be enabled only after you restart the Textures application.

HyperTeX Links

The Los Alamos HyperTeX package and the the LaTeX hyperref package are both supported in Textures 2.0; see the hyperTeX ReadMe on the CD for examples and more details. Local links within documents work. To take advantage of the external linking capability, you’ll need to run Internet Config, a helper application which will launch your preferred browser and/or email program in response to a click on an outside URL. You may already have Internet Config installed on your machine; if not, check the CD Extras folder for Internet Config 1.4.

Using the LATEX hyperref package

Textures 2.0 has support for the hyperref package. To use the hyperref package, simply include the following in the preamble of your document: `\usepackage{hyperref}` Here is a sample link, where Blue Sky is the text that will anchor the link:

```
\href{http://www.bluesky.com}{Blue Sky}
```

When you Command-click on a link, Textures will use Internet Config to open your preferred web browser. If you don't have Internet Config installed, you will need install it (it is on the Textures 2.0 CD in the `CD Extras` folder).

Textures 2.0 supports links within a document, links to URLs, and links to e-mail and FTP addresses. Textures 2.0 does not support PDF marks. Links to other Textures documents and color support for the hyperref package will be available in a subsequent release.

More information of the `hyperref` package can be found in the file `Hyperref guide`, which is in the `LaTeX Documentation` folder inside of the `LaTeX support` folder.

Synchronicity Tips and Troubleshooting

Synchronicity and older Machines

If you have compiled formats with 68K versions of Textures, it will be necessary to recompile (or "`\dump`") them to take advantage of the synchronicity features of Textures 2.0. 68K formats will function, but the Command-click, and Follow Focus will not be operational. 68K formats also cause Textures to run in 68K emulation mode, slowing down performance on PowerPC machines.

Custom 68K formats may be easily recompiled using the "Make Format" command in the File Menu, or using the older "dump" command (see page 111).

Synchronicity and Memory While Synchronicity will work correctly in a standard Textures partition of 4 to 6 MB, allocating enough memory to keep your entire document in memory can speed things up considerably.

“Stale” Links If you are making lots of editing changes to a document without typesetting, the crosslinking information becomes “stale” and will drift. Just typeset again to restore the correct correlations, or leave the document in Flash Mode while editing.

Turning Off Synchronicity and Follow Focus Turning off Synchronicity will create smaller documents, and turning off Follow Focus will speed up typesetting time.

Desktop Typography on the Macintosh

Textures, typesetting software for the Macintosh computer, integrates the extraordinary precision and range of the TEX typesetting language with the extraordinary responsiveness and advanced graphics capabilities of the Macintosh. In this section, we introduce the capabilities of TEX and survey the Textures system.

CONTENTS

TEX: A Work of Art

TEX, the typesetting engine of Textures, is an extremely powerful, versatile, programmable typesetting language created by Stanford University's Donald Knuth. Especially designed to automatically reproduce the quality of fine hand-set type, TEX is equally at home whether it is typesetting complex math and scientific notation to microscopic specifications, processing thousands of pages of mailing labels, or producing complex book-length or even multi-volume works.* From the ground up, TEX is built to last through indefinite future generations of hardware, and it is admirably suited to any task that requires precise typesetting and programmability.

Textures is a complete implementation of standard TEX with substantial added functionality.† In Textures, experienced TEX users will find the most versatile and easy-to-use implementation of TEX available on any machine. Users who are new to TEX will discover a uniquely capable typesetting system, giving you a level of control and flexibility completely unlike any other publishing system.

Compared to "traditional" TEX systems, Textures has the following distinctive advantages:

- Textures is an *integrated TEX system:* editing, viewing, and printing are combined in a single package.

- Textures is a *high-performance, high-capability TEX*. Textures' large memory model is capable of handling the largest conceivable projects. Key pieces of TEX have been rewritten in assembly language for lightning speed. "High-end" TEX capabilities such as virtual fonts are also supported.

* TEX (pronounced *tech*, as in "high-tech") is actually an upper-case tau, epsilon, and chi (τεχ), from the Greek word τεχνη *(technē)*, meaning art or craft, and the root of our words *technology*, *technique*, etc. It is cognate to the Latin *texere*, to weave or fit together into a complex structure; *textura* is the art or process of weaving or, by extension, a framework or structure. In Epicurean philosophy, *texture* means the proportion of atoms to void: the warp and weft of the universe.

† This means that Textures passes Donald Knuth's exacting TRIP test for full conformance to standard TEX.

- Textures provides a *unique user interface to TeX*. In `Flash mode`, you can view typeset output instantly on the screen and edit your TeX files interactively. Textures also lets you define your own macros as menu items, greatly increasing your efficiency. Linked together with customized formats, these `Macros` menus let you “package” specific typesetting applications together with a custom interface.

- Textures is *Macintosh integrated*. This means that PostScript fonts are fully supported and that Macintosh and PostScript graphics can be integrated into TeX files. It also means that you can export typeset TeX documents to Adobe Illustrator, making fine typography available to the Illustrator world.

But before looking at the specifics of the Textures system, let’s look in general terms at TeX itself.

Typesetting with TeX

TeX processes text-only source files to produce typeset output. These source files contain text and embedded TeX commands. Textures presents these files in Macintosh windows and enables you to enter TeX commands in a window (the **Edit window**) and immediately see the typeset results in another window (the **Typeset window**). (For an illustrated “quick start,” see Chapter 1.)

Unlike ordinary word-processing or desktop-publishing programs, TeX is entirely code-driven. Code-based document production encourages format design separate from writing the text. The logical design of a document style that TeX makes possible is especially well suited to production settings that include multiple authors, editors, graphic artists, programmers, and designers, as well as to very large projects such as reference manuals, catalogs, directories, and other database publishing. At the paragraph level, TeX automatically handles the subtleties of traditional typesetting with a precision impossible in hand-set type. But TeX is also designed for more complete structures, and you can program TeX to define formats for an entire book or set of books, thus letting you control the formatting of your documents from a global level.

The programmed layouts that you create with Textures are fully reusable. Once you have described how you want a document to look, that format can be used again as is, or employed as a base for more elaborate designs. It can also be saved in a compact binary form (called a TeX **format**) and then used as a transparent extension of standard TeX.

Computer Science and Classic Typography

Donald Knuth, mathematician and computer scientist, is considered by many to be the dean of computer science, above all for his multi-volume opus(-in-progress), *The Art of Computer Programming.* Professor Knuth is very exacting in the publication of his books. His first volume, *Fundamental Algorithms* (1968), was sent to Germany to be hand set by experienced typesetters who understood the subtleties of mathematical typesetting. But the men and women involved with Monotype machinery, who hand set each line of type using hot lead ingots, would not be around forever, and at the rate of one volume every five years, the hand-set ways would not last until the entire seven-volume set was finished.

Knuth tried computer-generated typesetting, but the results were so poor that he ended up setting aside his other projects to devote the next decade to the twin goals of creating the best possible digital typesetting system and making a software system that would be wholly independent of changes in printing technologies—when the future generations of typesetting machinery came out, he wanted to know that his typesetting system would continue to produce superior results. Knuth's interest in type design and the craft of typesetting became a passion, in which he enlisted world-reknowned practitioners of typography, and the resulting TeX system embodies the finest rules for typesetting not just mathematics, but all type. The result is a system that gives you not only programmability and automation, but *qualitatively* better results than traditional typography.

The Excellence of TeX

TeX combines the precision and control of handset type with programmable capabilities that don't exist on any other system.

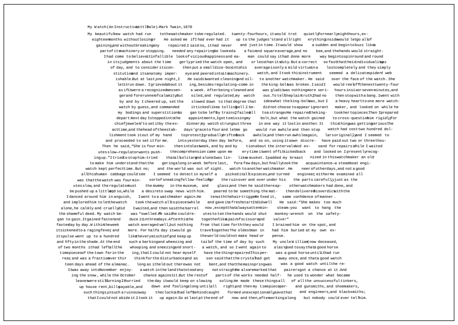
Virtually Unlimited Precision

TeX operates with electron-microscopic precision: it calculates text placement not in terms of points or quarter-points,* but in terms of internal units known as

* A **point** is a typographical unit of measure; there are 72.27 points to the inch, as defined by the National Bureau of Standards. (Here, compare TeX's strict adherence to the typograhic standard versus the ordinary

"RSU"s, which stands for Ridiculously Small Units: one RSU is about a hundred times smaller than the wavelength of visible light! (There are 65,536 RSUs in one printer's point.) For example, you can use TEX to typeset microfiche with complete accuracy. Compare this to actual output device resolutions—TEX is not likely to be made obsolete by advancing hardware!

Typographic Excellence TEX handles ligatures and kerning automatically and has an intricate mechanism for justifying lines; it also hyphenates words automatically when necessary. (TEX's hyphenation algorithm is so sophisticated that it in itself formed the thesis material for a graduate student of Professor Knuth's [see *The TEXbook*, Appendix H.]) TEX's optimal line breaking means more uniform spacing between words and an even, consistent look to the page: no "rivers" of white space down the page or clusters of hyphens breaking lines, but crisp, even areas of gray and white.



My Watch (An Instructive Little Tale)—Mark Twain, 1870

My beautiful new watch had run eighteen months without losing or gaining, and without breaking any part of its machinery or stopping. I had come to believe it infallible in its judgments about the time of day, and to consider its constitution and its anatomy imperishable. But at last, one night, I let it run down. I grieved about it as if it were a recognized messenger and forerunner of calamity. But by and by I cheered up, set the watch by guess, and commanded my bodings and superstitions to depart. Next day I stepped into the chief jeweler's to set it by the exact time, and the head of the establishment took it out of my hand and proceeded to set it for me. Then he said, "She is four minutes slow—regulator wants pushing up." I tried to stop him—tried to make him understand that the watch kept perfect time. But no; all this human cabbage could see was that the watch was four minutes slow, and the regulator must be pushed up a little; and so, while I danced around him in anguish, and implored him to let the watch alone, he calmly and cruelly did the shameful deed. My watch began to gain. It gained faster and faster day by day. Within the week it sickened to a raging fever, and its pulse went up to a hundred and fifty in the shade. At the end of two months it had left all the timepieces of the town far in the rear, and was a fraction over thirteen days ahead of the almanac. It was away into November enjoying the snow, while the October leaves were still turning. It hurried up house rent, bills payable, and such things, in such a ruinous way that I could not abide it. I took it to the watchmaker to be regulated. He asked me if I had ever had it repaired. I said no, it had never needed any repairing. He looked a look of vicious happiness and eagerly pried the watch open, and then put a small dice-box into his eye and peered into its machinery. He said it wanted cleaning and oiling, besides regulating—come in a week. After being cleaned and oiled, and regulated, my watch slowed down to that degree that it ticked like a tolling bell. I began to be left by trains, I failed all appointments, I got to missing my dinner; my watch strung out three days' grace to four and let me go to protest; I gradually drifted back into yesterday, then day before, then into last week, and by and by the comprehension came upon me that all solitary and alone I was lingering along in week before last, and the world was out of sight. I seemed to detect in myself a sort of sneaking fellow-feeling for the mummy in the museum, and a desire to swap news with him. I went to a watchmaker again. He took the watch all to pieces while I waited, and then said the barrel was "swelled." He said he could reduce it in three days. After this the watch averaged well, but nothing more. For half a day it would go like the very mischief, and keep up such a barking and wheezing and whooping and sneezing and snorting, that I could not hear myself think for the disturbance; and as long as it held out there was not a watch in the land that stood any chance against it. But the rest of the day it would keep on slowing down and fooling along until all the clocks it had left behind caught up again. So at last, at the end of twenty-four hours, it would trot up to the judges' stand all right and just in time. It would show a fair and square average, and no man could say it had done more or less than its duty. But a correct average is only a mild virtue in a watch, and I took this instrument to another watchmaker. He said the king-bolt was broken. I said I was glad it was nothing more serious. To tell the plain truth, I had no idea what the king-bolt was, but I did not choose to appear ignorant to a stranger. He repaired the king-bolt, but what the watch gained in one way it lost in another. It would run awhile and then stop awhile, and then run awhile again, and so on, using its own discretion about the intervals. And every time it went off it kicked back like a musket. I padded my breast for a few days, but finally took the watch to another watchmaker. He picked it all to pieces, and turned the ruin over and over under his glass; and then he said there appeared to be something the matter with the hair-trigger. He fixed it, and gave it a fresh start. It did well now, except that always at ten minutes to ten the hands would shut together like a pair of scissors, and from that time forth they would travel together. The oldest man in the world could not make head or tail of the time of day by such a watch, and so I went again to have the thing repaired. This person said that the crystal had got bent, and that the mainspring was not straight. He also remarked that part of the works needed half-soling. He made these things all right, and then my timepiece performed unexceptionably, save that now and then, after working along quietly for nearly eight hours, everything inside would let go all of a sudden and begin to buzz like a bee, and the hands would straightway begin to spin round and round so fast that their individuality was lost completely, and they simply seemed a delicate spider's web over the face of the watch. She would reel off the next twenty-four hours in six or seven minutes, and then stop with a bang. I went with a heavy heart to one more watchmaker, and looked on while he took her to pieces. Then I prepared to cross-question him rigidly, for this thing was getting serious. The watch had cost two hundred dollars originally, and I seemed to have paid out two or three thousand for repairs. While I waited and looked on I presently recognized in this watchmaker an old acquaintance—a steamboat engineer of other days, and not a good engineer, either. He examined all the parts carefully, just as the other watchmakers had done, and then delivered his verdict with the same confidence of manner.

He said: "She makes too much steam—you want to hang the monkey-wrench on the safety-valve!"

I brained him on the spot, and had him buried at my own expense.

My uncle William (now deceased, alas!) used to say that a good horse was a good horse until it had run away once, and that a good watch was a good watch until the repairers got a chance at it. And he used to wonder what became of all the unsuccessful tinkers, and gunsmiths, and shoemakers, and engineers, and blacksmiths; but nobody could ever tell him.

Figure 1. Optimal line breaks yield extraordinary typography.

Programmability Among publishing systems, TEX is uniquely programmable. This means that if your TEX system doesn't do precisely what you want, you can change it to do so. When you're finished, you will have a custom-tailored TEX that can produce beautiful documents *exactly* to your specifications. Thousands of macros already exist for doing things like typesetting journal articles, books (complete with indices, bibliographies, tables of contents), theses, catalogs, correspondence, and much more. (The most popular macros package, LATEX is included in Classic Textures.) The TEX system actually contains a special-purpose, but fully general programming language—a BASIC interpreter

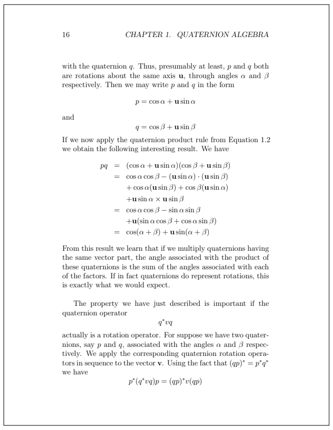
Macintosh world where points are reckoned at "about" 72 per inch.) Twelve points are equal to one pica. If you're not familiar with such typesetting terms, you can read Chapter 10 of *The TEXbook* or see the Glossary.

has even been written in the TEX language! And the size of jobs you can typeset is practically unlimited: with enough memory, you could typeset hundreds of thousands of pages (e.g., an entire encyclopedia) with TEX. (About 10K bytes are required per average page of text.)

A Portable Standard TEX is strictly standardized, and designed so that it will run on virtually any kind of computer. TEX currently runs on the full range of personal computers and workstations, including the IBM PC, Sun, NeXT, and Amiga; on mainframe computers such as IBM and DEC; and even on Cray supercomputers. TEX's universal machine base and rigid standardization mean that authors with completely different computer architectures can publish in TEX and know that their TEX files can be typeset with absolutely consistent results on any other machine. This encourages collaboration of researchers and authors in a way not imagined before, and books having dozens of contributing authors from all over the world have been published with TEX. This type of portability cannot be found on any other typesetting or word-processing system.

International Access Like the Macintosh, TEX is designed to be language-independent. All you need to use French or German, for instance, is another set of hyphenation patterns, since the rules regarding where to break a word vary from language to language. (The "CD Extras" folder on the "Textures CD" contains a wide variety of foreign language hyphenation patterns.). And with the international TEX groups that are set up, there is abundant information about how to typeset in many different languages. In fact, TEX has produced a worldwide following with no fewer than a dozen user groups around the globe, including GUTenberg in France, TUG in the U.S. and U.K. TUG in Britain, and DANTE in Germany, with several thousand members.

Standard Formats Numerous prepackaged document formats, such as thesis styles for major universities, have already been created for TEX. This means that you can focus on the content of your document, and not the style—you only need to run TEX with the appropriate package (or macro) set to produce your properly formatted document. Publishers of technical books routinely provide authors with TEX formats into which they can "pour" their text.



16 *CHAPTER 1. QUATERNION ALGEBRA*

with the quaternion q. Thus, presumably at least, p and q both are rotations about the same axis $\mathbf{u}$, through angles α and β respectively. Then we may write p and q in the form

$$p = \cos\alpha + \mathbf{u}\sin\alpha$$

and

$$q = \cos\beta + \mathbf{u}\sin\beta$$

If we now apply the quaternion product rule from Equation 1.2 we obtain the following interesting result. We have

$$\begin{aligned} pq &= (\cos\alpha + \mathbf{u}\sin\alpha)(\cos\beta + \mathbf{u}\sin\beta) \\ &= \cos\alpha\cos\beta - (\mathbf{u}\sin\alpha)\cdot(\mathbf{u}\sin\beta) \\ &\quad + \cos\alpha(\mathbf{u}\sin\beta) + \cos\beta(\mathbf{u}\sin\alpha) \\ &\quad + \mathbf{u}\sin\alpha \times \mathbf{u}\sin\beta \\ &= \cos\alpha\cos\beta - \sin\alpha\sin\beta \\ &\quad + \mathbf{u}(\sin\alpha\cos\beta + \cos\alpha\sin\beta) \\ &= \cos(\alpha+\beta) + \mathbf{u}\sin(\alpha+\beta) \end{aligned}$$

From this result we learn that if we multiply quaternions having the same vector part, the angle associated with the product of these quaternions is the sum of the angles associated with each of the factors. If in fact quaternions do represent rotations, this is exactly what we would expect.

The property we have just described is important if the quaternion operator

$$q^*vq$$

actually is a rotation operator. For suppose we have two quaternions, say p and q, associated with the angles α and β respectively. We apply the corresponding quaternion rotation operators in sequence to the vector $\mathbf{v}$. Using the fact that $(qp)^* = p^*q^*$ we have

$$p^*(q^*vq)p = (qp)^*v(qp)$$

Figure 2. Standardized journal article format in LATEX.

TEX's Limits

TEX is a production tool rather than a visually oriented *design* tool. But it does provide a design *environment*: a set of capabilities that allow you to achieve more sophisticated results automatically than you probably thought possible. For example, knowing that you have the capabilities of TEX at your disposal, you can define the location of text and harmonies of light and dark on the page exactly to your specifications, and instantly see and adjust output once your basic design is established. Graphic designers have thus employed TEX to produce results that would be otherwise impossible. (An example is shown in Figure 3.)

TEX also requires some commitment and rewards a serious investment. If you are using TEX for the first time, it will probably take a few day's effort to begin producing high-quality output that you are happy with. TEX has lots of personality built in, and at times, TEX may seem to be just plain ornery. But the control you will gain over your results will repay the effort, and you will discover in TEX a system of unlimited depth.

Figure 3. Graphic design with TEX: here we've created a format such that pages of varying length were each typeset in the proportions of a golden rectangle. (*Life Cast*, by Willa Shalit; design by Principia Graphica, published by Beyond Words, Portland, Oregon.)

The Textures System

TEX is the programmable core of Textures. As we've mentioned, Textures also includes a number of facilities not found in more traditional TEX environments (Figure 4). These are:

- A built-in editor with a user-defined macro menus capability.
- Display capabilities that let you closely preview your typeset output. (In the instantaneous Flash mode, the results of your TEX commands are immediately reflected in your typeset output as you type them.)
- A picture manager for including Macintosh graphics in your documents.
- Printing facilities.
- Font management (via the EdMetrics tool) and built-in support for all of the standard PostScript fonts.
- Computer Modern PostScript fonts.

With the exception of the EdMetrics tool, these functional pieces are completely integrated and won't appear as separate pieces.

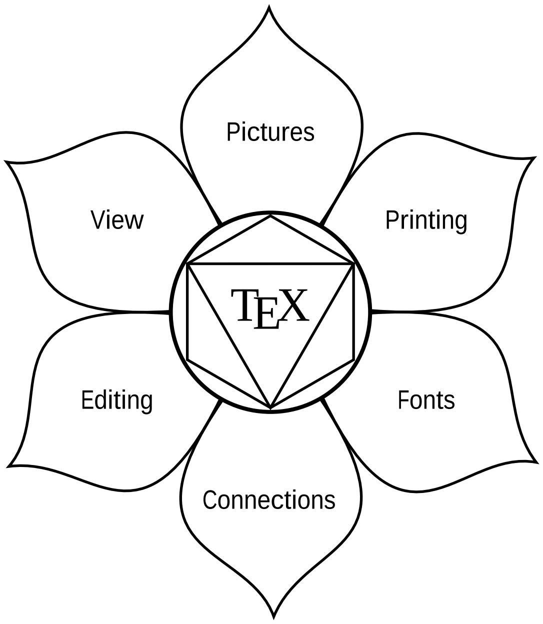


Figure 4. The Textures system.

To orient you to the physical pieces of the system, we'll now take a look at the files and folders you received with your distribution disks.

A Textures Roadmap

Here, we briefly survey the different pieces that form the central and outlying parts of the Textures domain. At this point, we're only assigning some names to the places: you don't need to know what all of these are now, and some things you may never need to know. Fuller details concerning your distribution disks and files can be found in your *Textures Installation Guide* and in Appendix A of this manual, beginning on page 197.

Textures Files and Folders Assuming you have already installed Textures, you'll see the following files and folders in the Textures folder:

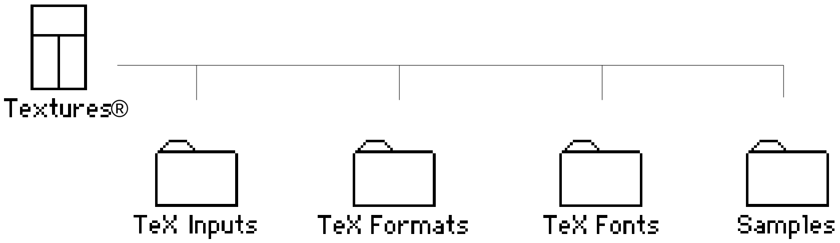


Figure 5. Textures files and folders.

Samples

To illustrate some basic Textures operations, we have provided several sample documents with your Textures package. You can examine these files on your screen, typeset them, and print them to check the results. These files can also serve as models to follow when you format your own documents.

The “Sample1” file will introduce you to some standard features of Plain TeX. “Pictures Example” shows some ways of integrating pictures into Textures documents. See page 202 for a survey of the sample files provided with Textures.

TeX Formats

The TeX Formats folder contains “canned” **TeX formats** created by yourself or others. A TeX format is a collection of TeX commands to be used when you typeset a document. A format may be anything from a few commands to supplement the default Plain TeX format to a completely different TeX world, such as LaTeX. See Chapter 5, “Macros and Formats,” beginning on page 100, for information on TeX formats.

TeX Inputs

The TeX Inputs folder stores Textures documents and illustrations that you intend to include in another Textures document. See page 67 for more information.

TeX Fonts

The TeX Fonts folder contains **metric information** for fonts that you use with TeX. TeX looks in this folder to find the information it needs on character dimensions, kerning, and ligatures. (Though metric information for most of the standard Macintosh and TeX fonts is actually built into the Textures application itself.) The TeX Fonts folder can also contain fonts themselves, in which case they are available only to Textures but not to other programs. See Chapter 7, “Fonts in Textures,” beginning on page 134, for information on fonts and font metrics.

Types of Textures Files

Textures creates several kinds of files; they are merely introduced here, with full explanation deferred to later chapters.

Textures text files: these are the standard TeX files containing your input text and TeX formatting commands.

Textures DVI files: these are files containing the typeset portion of Textures files. These files may be read and printed by Textures Reader or used to exchange typeset files with TeX on other computer systems.

“Canned” format files: these contain TeX formats in a compact, “predigested” binary form.

Font metrics “suitcase” files: these contain precise descriptive information about fonts in the format that TeX requires.

As far as Textures is concerned, there are no restrictions on how you name your files. But TeX itself will only read one-word filenames (i.e., it regards a space character as the end of the name), so if you are writing a Textures file that may be input into another TeX document, leave out the spaces.

Documentation

The present guide explains how to use Textures but does not explain how to use the TeX language itself. For that, you’ll want to refer to *The TeXbook* and to the books listed below. *The TeXbook* is the definitive guide to TeX written by its creator, Donald Knuth. A description of every TeX command (or **control**

sequence) can be found in it somewhere.* However, *The TEXbook* is not structured as a reference manual and can be quite opaque to the beginner.

Once you have gone through Chapters 1 and 2 of the present guide and familiarized yourself with some of the basics of using TEX, you may find it very useful to read Chapters 1–6 of *The TEXbook* for the exact details of TEX's operation. (Happily, many of the ugly operational details of running TEX presented in Chapter 6 of *The TEXbook* do not apply to running Textures on the Macintosh.)

For beginners to TEX or for easy reference, we highly recommend the following books:*

■ Raymond Seroul & Silvio Levy, *A Beginner's Book of TEX*, Springer-Verlag, 1991.

■ Arvind Borde, *TEX by Example*,

■ Paul W. Abraham, *TEX for the Impatient*, Addison-Wesley, 1990.

Those who are using the LATEX macro sets should select one of the following books as a primary reference book:

■ Leslie Lamport, *LATEX, A Document Preparation System (Second edition)*, Addison-Wesley, 1994.

■ Goossens, Mittelbach, and Samarin, *The LATEX Companion*, Addison-Wesley, 1994.

■ AMS-TEX book: George Grätzer, *Math into LATEX*, American Mathematical Society and Addison-Wesley.

For an introduction to LATEX, see page 108. For a detailed TEXnical bibliography, see Appendix D.

We'll now look at the basic functioning of the Textures system.

* For the last extensions to TEX, see also Knuth's document "TeX 3.0," in the Documents folder.

* These books can all be ordered from Blue Sky Research. See page 224 for details.

chapter

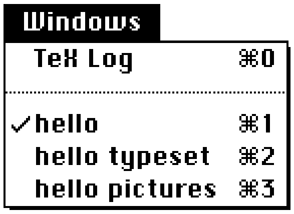
1

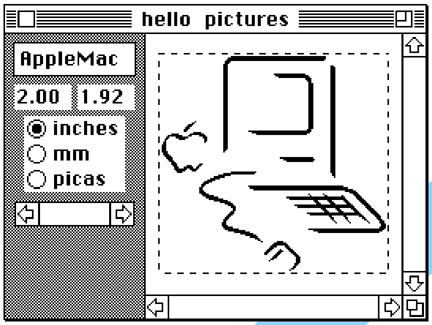
Basics

This chapter gives a very brief, visual overview of the basics of typesetting a document in Textures. A complete description of the process is given in Chapter 2, “Expanded Basics.”

Windows

A Textures document has four windows associated with it.



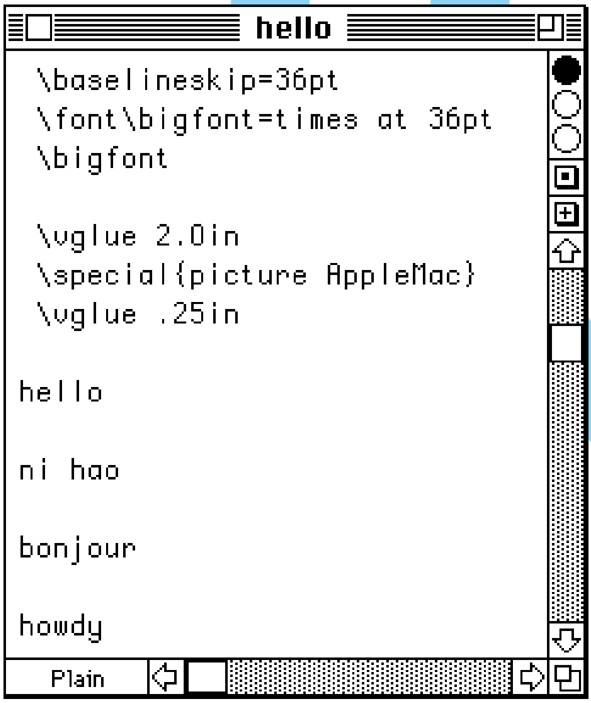


AppleMac

2.00 1.92

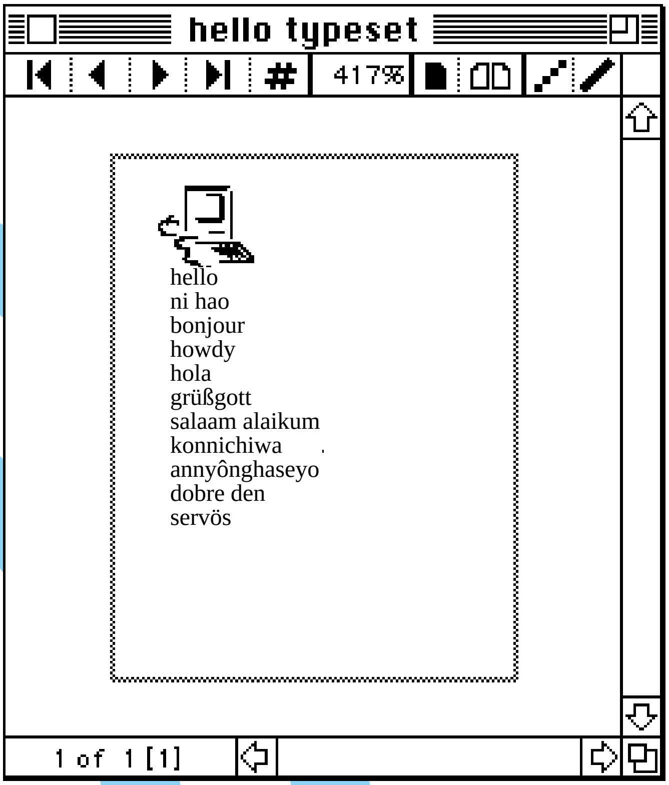
● inches
○ mm
○ picas

Pictures window



```
 \baselineskip=36pt
 \font\bigfont=times at 36pt
 \bigfont

 \vglue 2.0in
 \special{picture AppleMac}
 \vglue .25in

hello

ni hao

bonjour

howdy
```

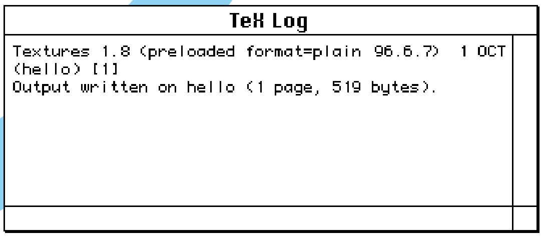
Edit window



hello
ni hao
bonjour
howdy
hola
grüßgott
salaam alaikum
konnichiwa
annyônghaseyo
dobre den
servös

Typeset window

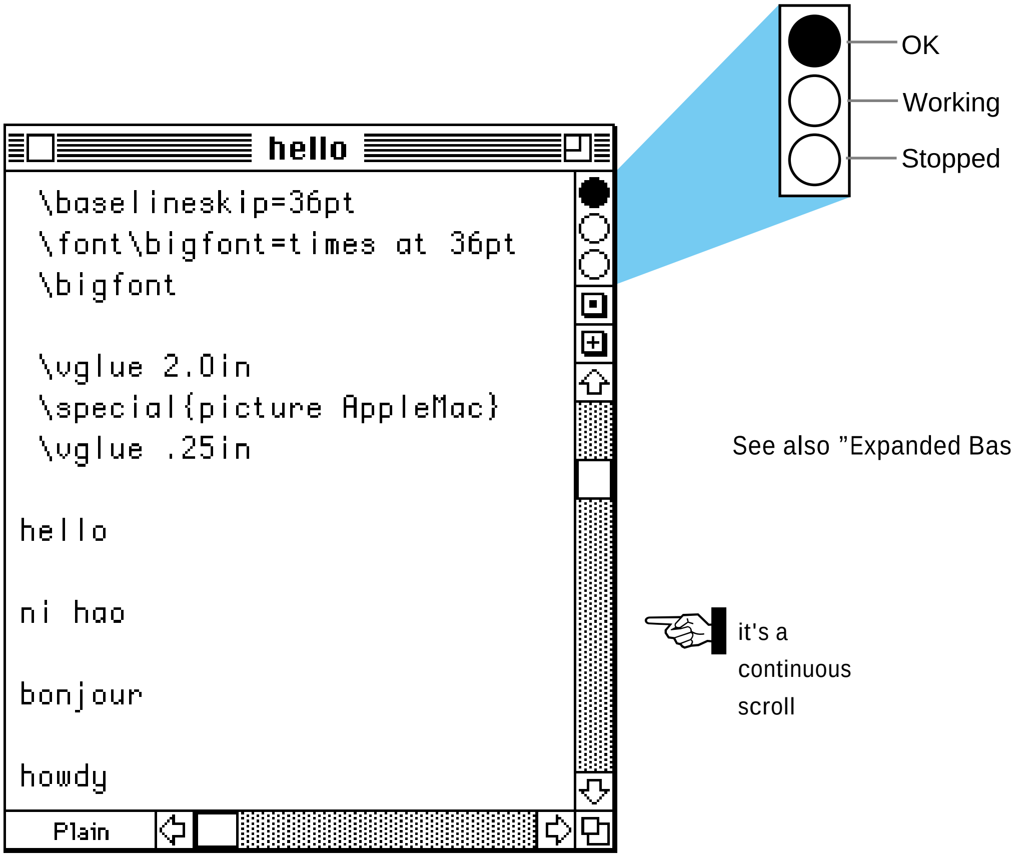
See also ”Expanded Basics“, p. 42



```
Textures 1.8 (preloaded format=plain 96.6.7)  1 OCT
(hello) [1]
Output written on hello (1 page, 519 bytes).
```

Log window

Edit window



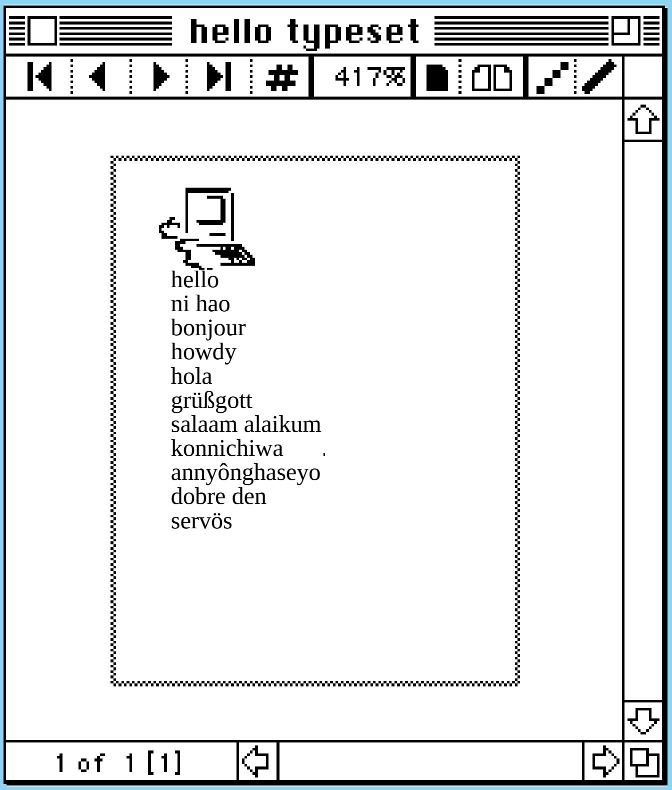
```
  \baselineskip=36pt
  \font\bigfont=times at 36pt
  \bigfont

  \vglue 2.0in
  \special{picture AppleMac}
  \vglue .25in

hello

ni hao

bonjour

howdy
```

See also "Expanded Basics", p. 47

☞ it's a continuous scroll

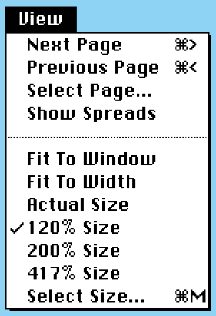
The Edit window is where you enter and edit your document's text together with your TEX typesetting commands. In the default Flash mode, TEX typesets text immediately as you enter it.

Typeset window



hello
ni hao
bonjour
howdy
hola
grüßgott
salaam alaikum
konnichiwa
annyônghaseyo
dobre den
servös

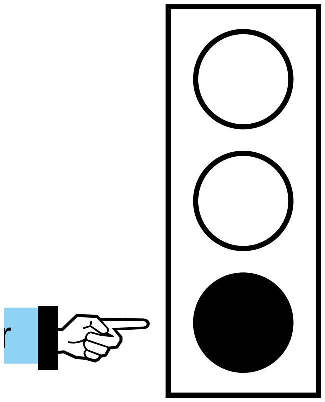
The Typeset window
presents your typeset results.



use the
View menu
to move
from page
to page

See also ”Expanded Basics“, p. 51

If you see the problem:



setting!

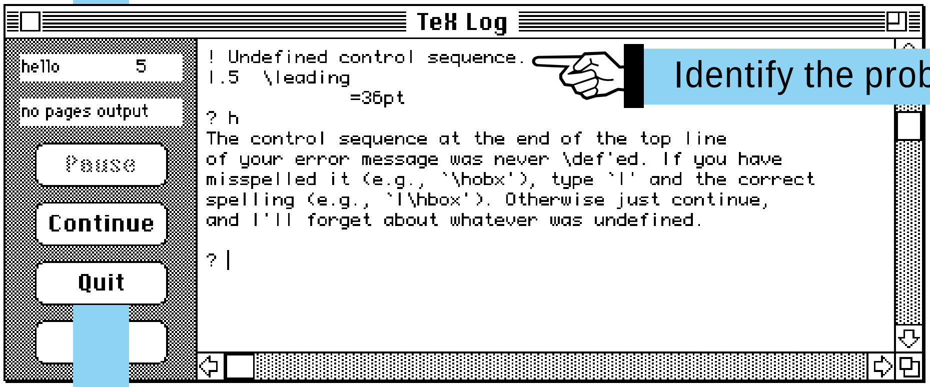
WHY?

TEX Log

The TEX Log chronicles the typesetting process. You can interact with the TEX Log window to detect and correct problems with your typesetting job.

See also "Expanded Basics", p. 48

Check the TEX Log:



```
! Undefined control sequence.
l.5  \leading
             =36pt
? h
The control sequence at the end of the top line
of your error message was never \def'ed. If you have
misspelled it (e.g., `\hobx'), type `I' and the correct
spelling (e.g., `I\hbox'). Otherwise just continue,
and I'll forget about whatever was undefined.

? |
```

Identify the problem

"Quit" returns you to the Edit window

Correct

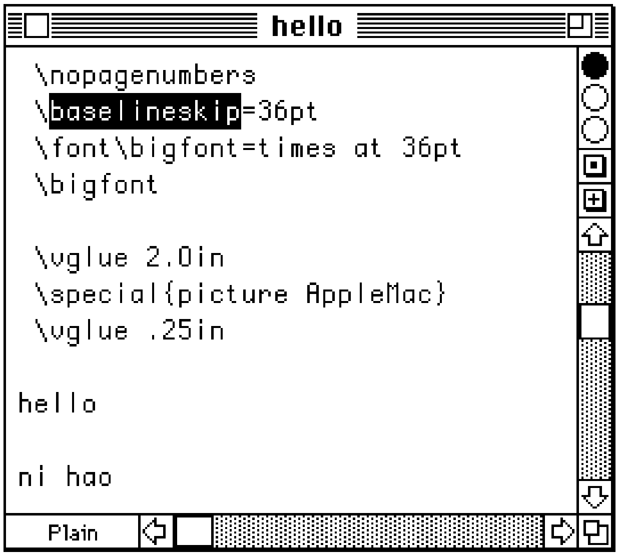
the

problem

in the

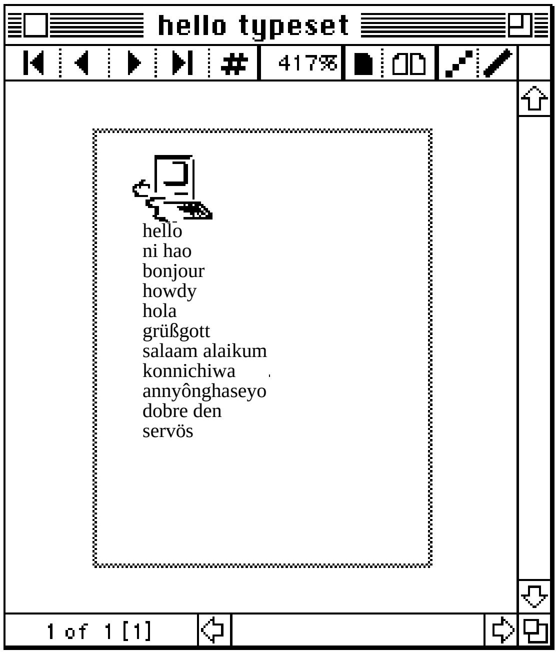
Edit

window



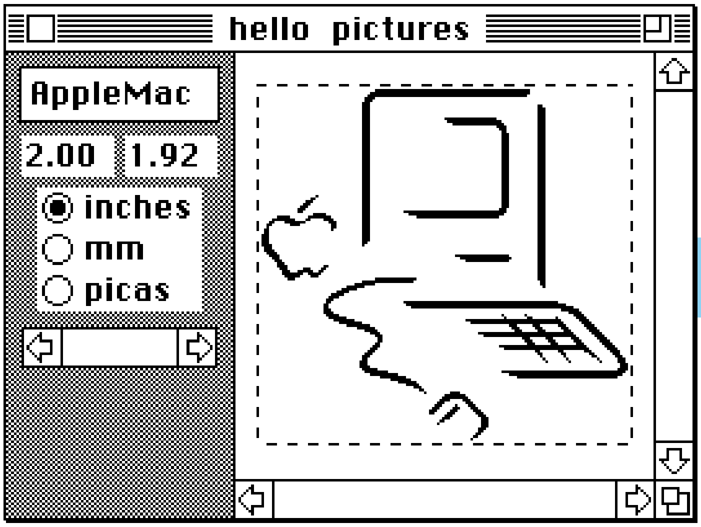
```
 \nopagenumbers
 \baselineskip=36pt
 \font\bigfont=times at 36pt
 \bigfont

 \vglue 2.0in
 \special{picture AppleMac}
 \vglue .25in

hello

ni hao
```

Successful typeset



hello
ni hao
bonjour
howdy
hola
grüßgott
salaam alaikum
konnichiwa
annyônghaseyo
dobre den
servös

Picture window



hello pictures

AppleMac

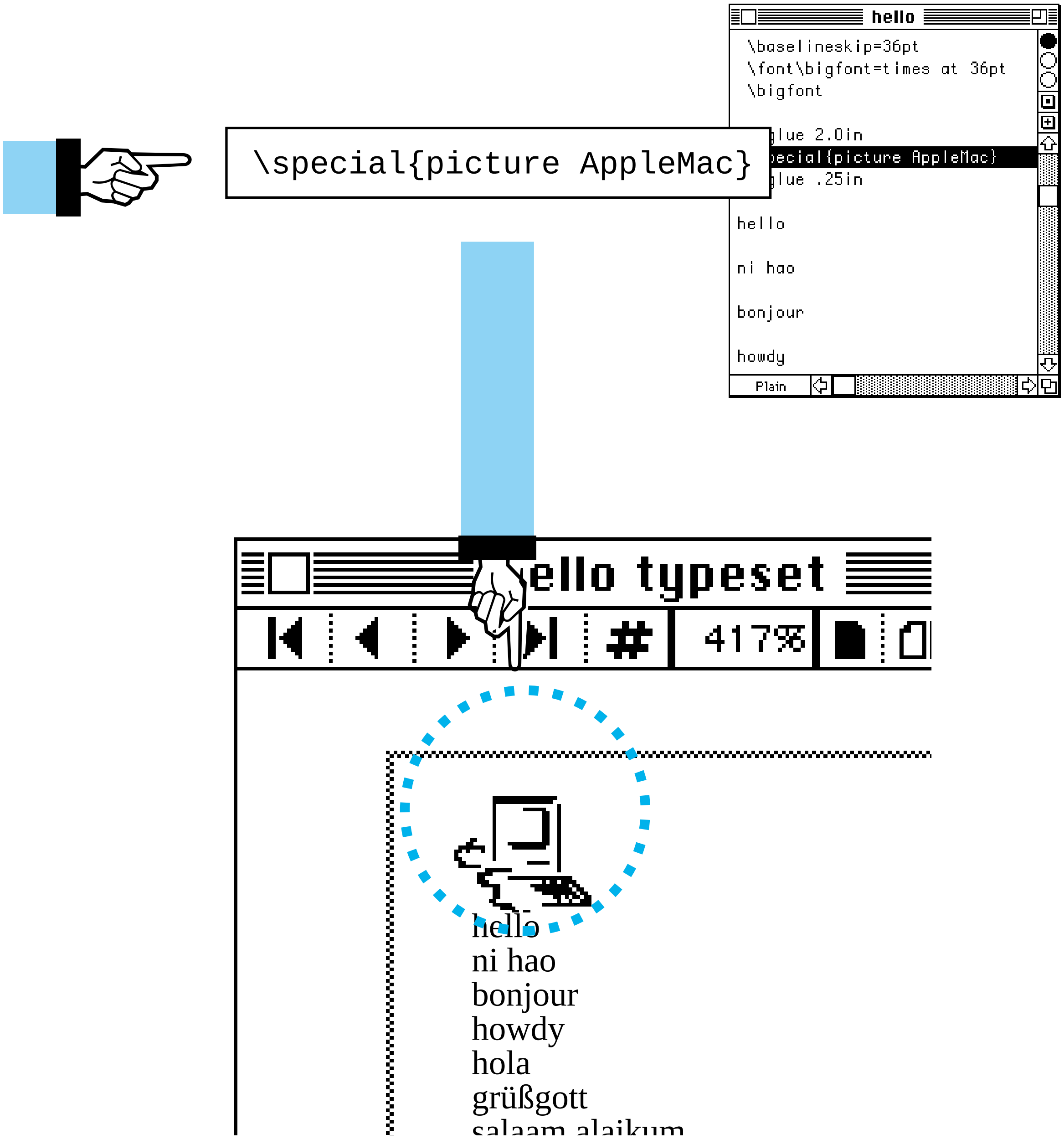
2.00 1.92

● inches
○ mm
○ picas

You can store Macintosh graphics in the document’s Pictures window, or store them in separate files. For example, you can store high-quality graphics in encapsulated PostScript (EPS) files.

See also ”Pictures“, p. 130

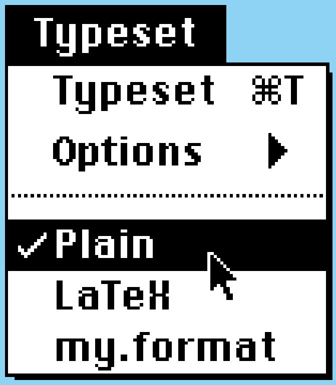
You include these pictures in your document with Textures’ special picture placement commands.



```
\special{picture AppleMac}
```

```
\baselineskip=36pt
\font\bigfont=times at 36pt
\bigfont

glue 2.0in
pecial{picture AppleMac}
glue .25in

hello

ni hao

bonjour

howdy
```

hello
ni hao
bonjour
howdy
hola
grüßgott
salaam alaikum



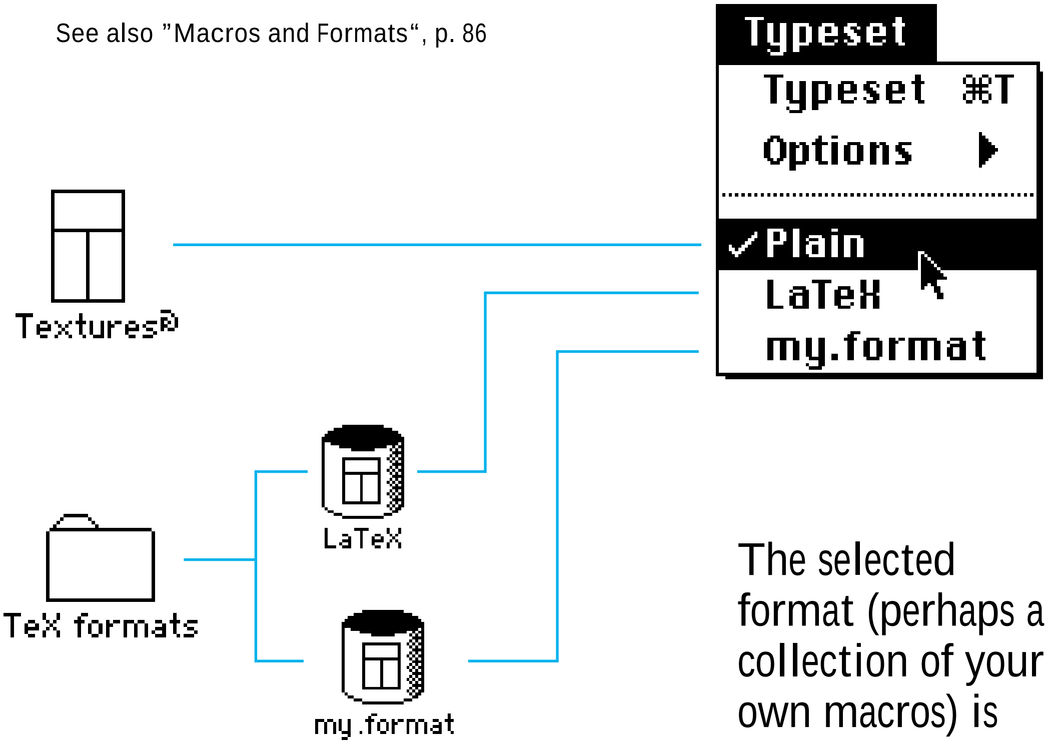
plain

Choosing the "Plain" format from the Typeset menu will provide you with "vanilla" default TeX. You have access to all of the capabilities of the TeX typesetting language.

See also ”Macros and Formats“, p. 86

format

See also ”Macros and Formats“, p. 86

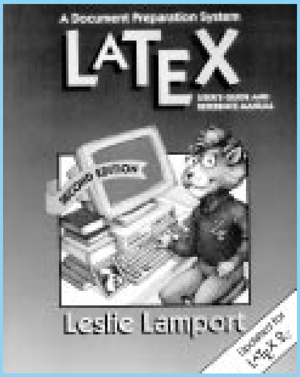


The selected format (perhaps a collection of your own macros) is automatically read into TEX each time you typeset. This format is the “application” that actually typesets your document.

Format files are kept in the TeX formats folder

reference

The TEXbook is the reference to the Plain TEX format.


A Document Preparation System
LATEX
User's Guide and Reference Manual
Second Edition
Leslie Lamport
Updated for LATEX 2ε

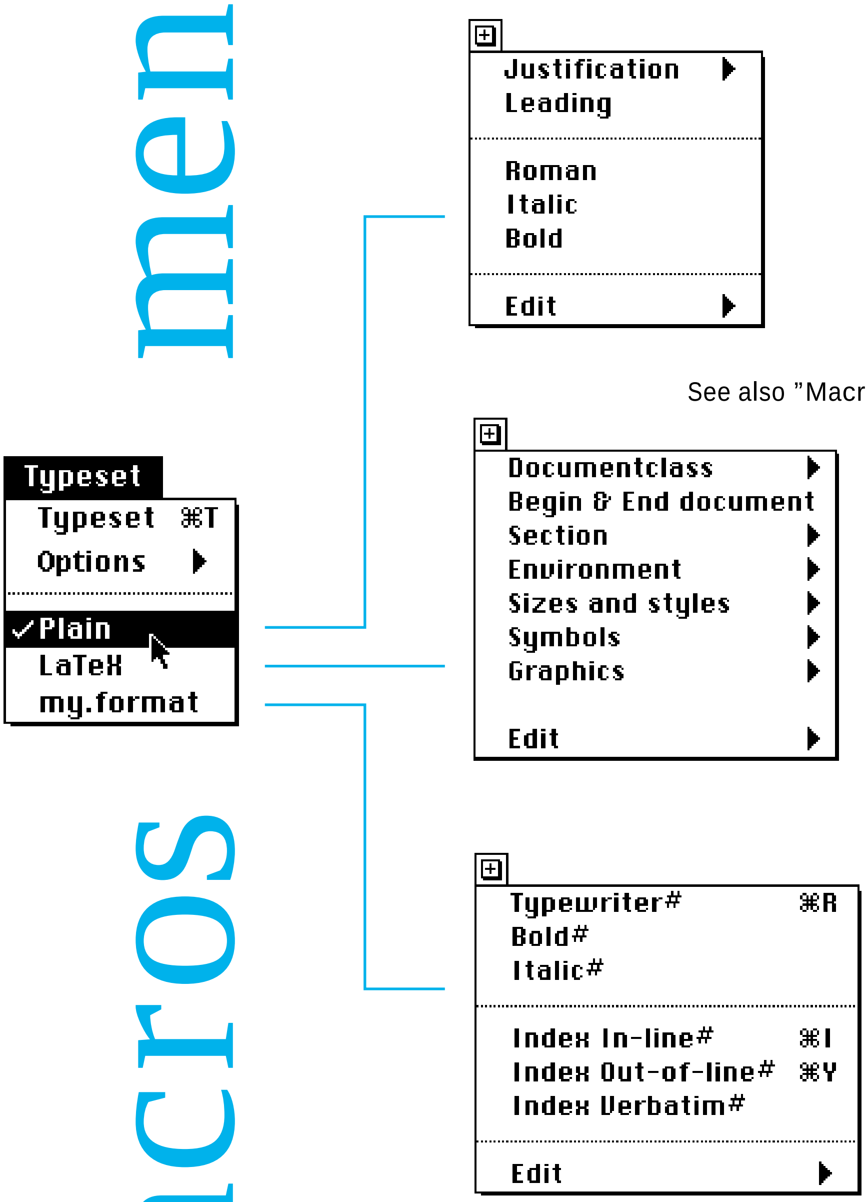
Other formats have their own reference books.

The Joy of TEX
A Gourmet Guide to Typesetting with the AMS-TEX macro package
M. D. SPIVAK, Ph.D.

See also ”For More Information“, p. 194

Macros menu

Each format has its own Macros menu.



See also ”Macros Menu“, p. 74

Use the Macros menu to store your frequently used macros as menu items.

chapter 2

Expanded Basics

This chapter describes the basic workings of Textures: how to create, edit, typeset, and print a Textures document. It takes you through the typesetting process by referring to sample files in the Textures “Samples” folder.

CONTENTS

The “Sample1” document in the Samples Folder illustrates some basic features of the TeX typesetting language. “The Frog King” is a sample LaTeX file. If you open one of the sample files now, you’ll be able to usefully refer to the points discussed in the rest of this chapter.

Windows attached to a Textures Document

In Chapter 1 we said that a document has four windows associated with it. To be more precise, every Textures document has *three* windows attached to it—an Edit window, a Typeset window, and a Pictures window (Figure 2-1). A fourth window, the TeX Log window is not attached to any particular document, but simply chronicles the most recent typesetting job. When you open a document in Textures, its attached windows will be opened along with it:

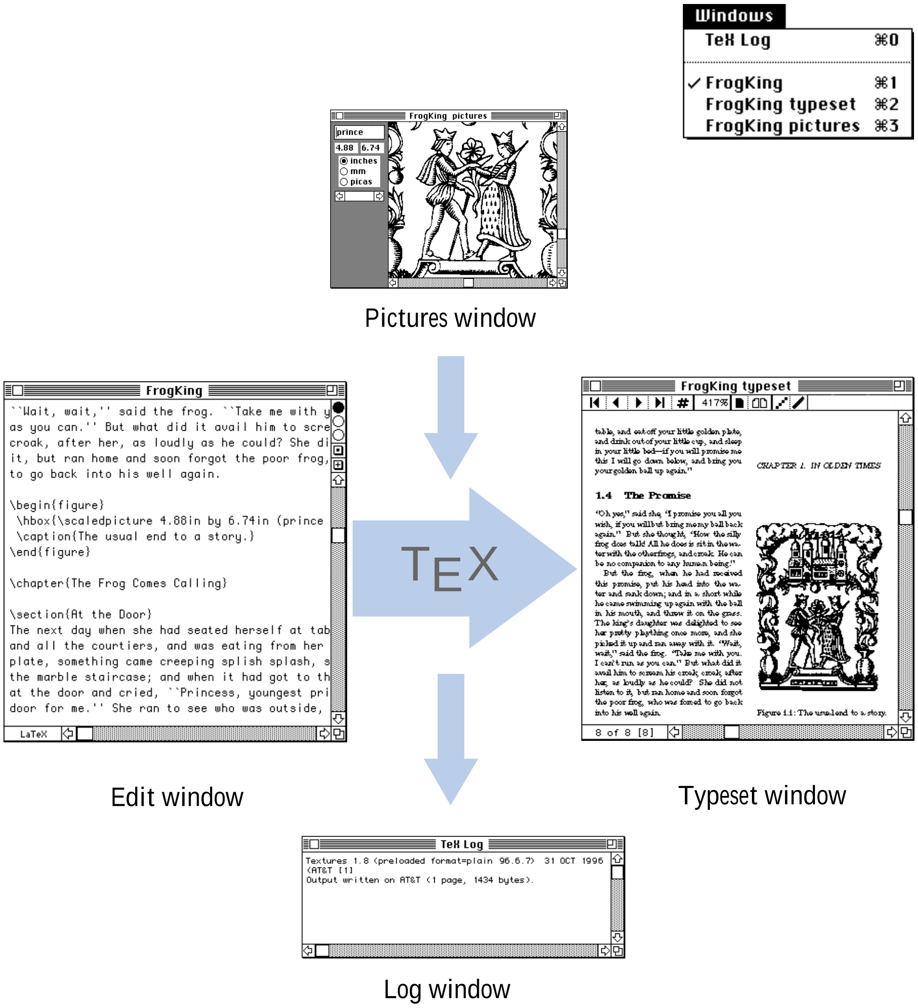
- The **Edit window** is the document’s master window, the window that first appears when you open a Textures document. It contains the document’s text together with the TeX commands that specify how it should be typeset. This is where all editing occurs.

- After TeX typesets your document, the document’s **Typeset window** displays the result. This is the window you print to produce typeset output. What you see in this window is what you will get when you print (but see the discussion of “Visual Quality” on page 65); and no TeX commands or other special symbols should normally appear in this window.

- The document’s **Pictures window** provides storage for any Macintosh graphics you want to insert into a Textures document. See Chapter 8, “Pictures,” beginning on page 152 for details.

The TeX **Log window** also appears whenever errors occur while you typeset a document with the `Typeset` command. (It stays out of sight in Flash mode unless you bring it to the front.) The Log window displays status and error messages and lets you pause TeX and provisionally correct errors encountered during typesetting.

Each of these windows is described in detail later in this chapter.



```
``Wait, wait,'' said the frog. ``Take me with y
as you can.'' But what did it avail him to scre
croak, after her, as loudly as he could? She di
it, but ran home and soon forgot the poor frog,
to go back into his well again.

\begin{figure}
 \hbox{\scaledpicture 4.88in by 6.74in (prince
 \caption{The usual end to a story.}
\end{figure}

\chapter{The Frog Comes Calling}

\section{At the Door}
The next day when she had seated herself at tab
and all the courtiers, and was eating from her
plate, something came creeping splish splash, s
the marble staircase; and when it had got to th
at the door and cried, ``Princess, youngest pri
door for me.'' She ran to see who was outside,
```

table, and eat off your little golden plate, and drink out of your little cup, and sleep in your little bed—if you will promise me this I will go down below, and bring you your golden ball up again."

CHAPTER 1. IN OLDEN TIMES

1.4 The Promise

"Oh yes," said she, "I promise you all you wish, if you will but bring me my ball back again." But she thought, "How the silly frog does talk! All he does is sit in the water with the other frogs, and croak. He can be no companion to any human being."

But the frog, when he had received this promise, put his head into the water and sank down; and in a short while he came swimming up again with the ball in his mouth, and threw it on the grass. The king's daughter was delighted to see her pretty plaything once more, and she picked it up and ran away with it. "Wait, wait," said the frog. "Take me with you. I can't run as you can." But what did it avail him to scream his croak, croak, after her, as loudly as he could? She did not listen to it, but ran home and soon forgot the poor frog, who was forced to go back into his well again.

Figure 1.1: The usual end to a story.

```
Textures 1.8 (preloaded format=plain 96.6.7)  31 OCT 1996
(AT&T [1]
Output written on AT&T (1 page, 1434 bytes).
```

Figure 2-1. Windows attached to a Textures document.

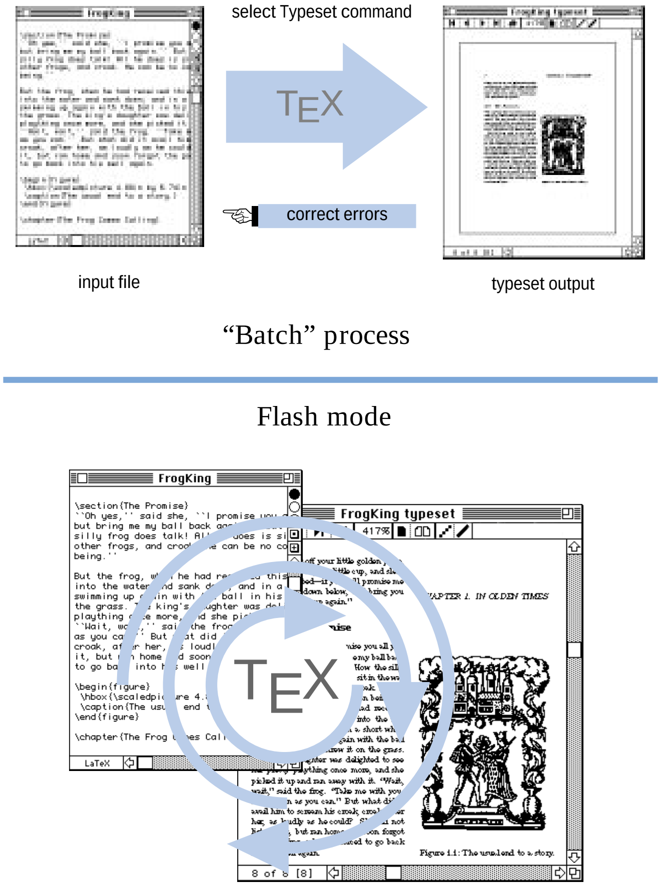
The Windows menu is a handy way to bring a window to the front. For any document that you are working on, you'll always see three windows listed in the Windows menu; for example, "FrogKing," "FrogKing typeset," and "FrogKing pictures," as well as the "TEX log" window. A check mark indicates the currently active window. You can open any number of Textures documents with their attached windows at the same time (limited only by available RAM).

Typesetting a Document

2.0 UPDATE: See new Synchronicity Features on page 15.

As we indicated in the Introduction, TeX is the “typesetting engine” of Textures, processing input text and producing typeset output. With Textures, you have two ways of telling TeX to typeset a document. In **Flash mode** (the initial default), TeX typesets your document as soon as you enter any text or formatting commands in the Edit window (Figure 2-2). Alternatively, you can turn Flash mode off and defer typesetting until you explicitly select the Typeset command. Each of these modes is suited to different tasks and to different working styles, and you will likely find that you use both of them.

When TeX begins to typeset a document, it also starts the TeX log, which displays information about the processing of your input, including error messages. The traffic signal will also blink during TeX processing.



“Batch” process

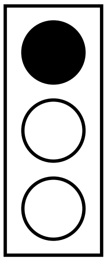
Flash mode

Figure 2-2. Flash mode and the Typeset command.

Flash Mode

2.0 UPDATE: See the new Follow Focus feature on page 16.

In Flash mode, TEX typesets your document continuously and any changes you make in the Edit window are reflected immediately in the Typeset window. When Flash mode is active, the **“traffic signal”** icon will be activated in the Edit window’s upper-right corner:



The upper, “green” light means that typesetting is complete.*

The center, “amber” light flashes when TeX is processing your document. (It flashes continually when you are typing.) It also flashes when TeX is paused because of an error (see page 57).

The lower, “red” light means that TeX has stopped typesetting due to a serious error.

Because it displays the effects of your formatting commands as you enter them, Flash mode enables a highly interactive working style and is the most powerful tool available for learning TeX quickly. When working in Flash mode, you may find it convenient to place a smaller Edit window on top of a full-screen Typeset window so that you can directly see your results as you enter or edit text.

On the other hand, when you are simply entering the text of a document, you may not need Flash mode. Flash mode can be used with large documents but is dramatically faster with small ones, and you may find it unacceptably slow when you are typesetting long documents on an older 68000-based Macintosh such as a Macintosh Plus or SE. See the “Tips for Working in Flash Mode” on page 191 for more information.

NOTE: Flash mode requires a memory partition of at least 2 megabytes (MB); 3 to 4 MB may be needed for very complex documents. (To change the Textures memory partition, quit Textures and use the Finder’s Get Info command on the Textures application.)

2.0 UPDATE: Memory requirements have changed with Sychronicity. See page 20.

To turn Flash mode on and off, select the `Flash mode` menu item. When Flash mode is off, your document will be typeset only when you explicitly select the `Typeset` command or click on the traffic signal.

* This is on the model of a railroad signal: the upper, green light means that the line is clear, and the lower, red light means that the line is blocked.

NOTE: When flash mode is off, the traffic signal will still blink during typesetting, but otherwise it will be blank.

NOTE: There is one TEX operation that creates a new format, `\dump`, which cannot be performed in Flash mode (see page 111).

The Typeset Command

The `Typeset` command (T on the keyboard) tells TEX to typeset the document in the active window. It also starts the TEX log. (You can also typeset the document by clicking on the traffic signal.) TEX will pause typesetting if it encounters an error, and the TEX Log window will come to the fore to ask you how to proceed (see page 57).

TEX Formats

The lower part of the `Typeset` menu contains one or more format choices that let you apply a variety of pre-defined formats to the typesetting process. A **TEX format** is a package of TEX commands that defines a particular "typesetting processor"—this could be anything from your own document style to a complete restructuring of the standard "Plain TEX" format. Each format is associated with its own `Macros` menu, which lets you specify your own menu items to go with that format (Figure 2-3). The currently selected format is indicated in the menu with a check mark.

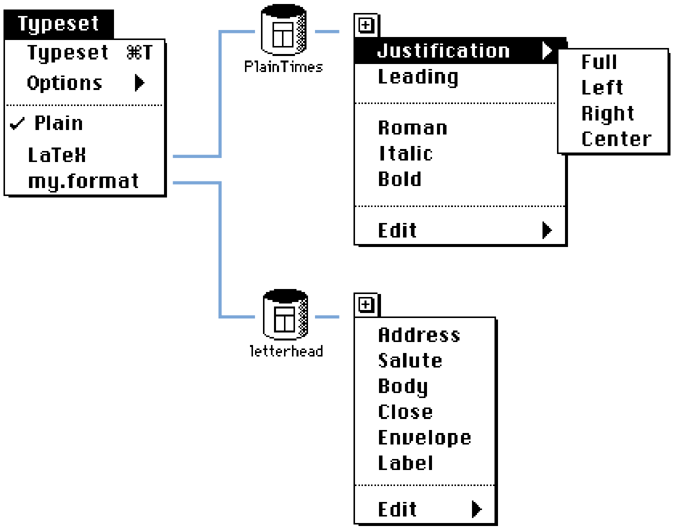


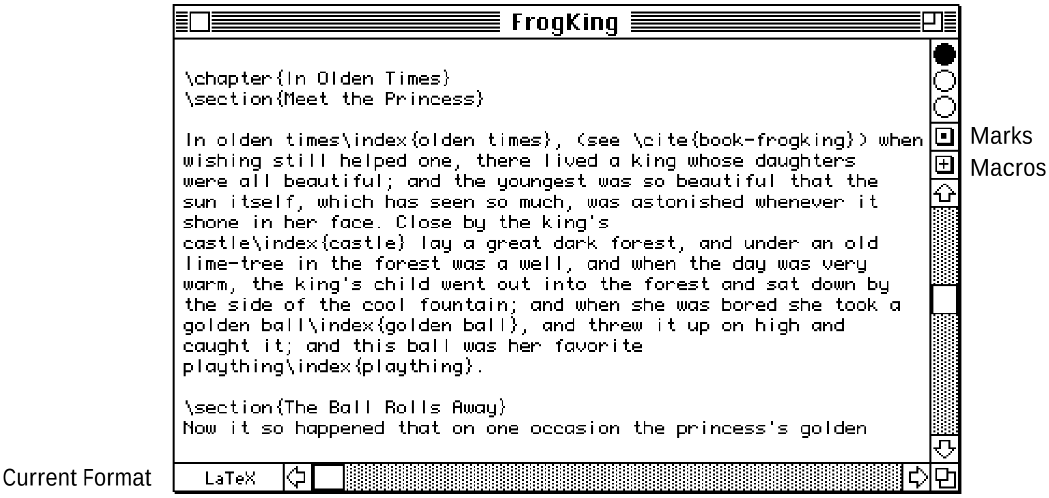
Figure 2-3. Predefined formats and Macros menus.

Textures lets you store a format in a precompiled or "canned" format file. You can create your own format with `Make Format` or TEX's `\dump` command. The most widely used predesigned format, LaTeX is included in the Classic Textures package. LaTeX, in turn, makes use of numerous **packages** such as AMS-Latex, which function as “sub-formats” for specific uses.

For typesetting the sample files, you should stay with the default `Plain` format (unless, of course, you are typesetting the LaTeX sample files). The **Plain TeX** format and formats in general are discussed in Chapter 5, “TeX Formats,” beginning on page 100.

Text Input: the Edit Window

The Edit window is the master window for any Textures document (Figure 2-4). It shows the text-only file that contains the document’s text together with the **TeX commands** (or, **control sequences**) that tell TeX how to format the typeset output. You’ll notice that each TeX command begins with an **escape character**, ordinarily the **backslash** (\) character. Command arguments and grouping are specified within **braces** ({ }).* The rest is the document’s text.



```
\chapter{In Olden Times}
\section{Meet the Princess}

In olden times\index{olden times}, (see \cite{book-frogking}) when
wishing still helped one, there lived a king whose daughters
were all beautiful; and the youngest was so beautiful that the
sun itself, which has seen so much, was astonished whenever it
shone in her face. Close by the king's
castle\index{castle} lay a great dark forest, and under an old
lime-tree in the forest was a well, and when the day was very
warm, the king's child went out into the forest and sat down by
the side of the cool fountain; and when she was bored she took a
golden ball\index{golden ball}, and threw it up on high and
caught it; and this ball was her favorite
plaything\index{plaything}.

\section{The Ball Rolls Away}
Now it so happened that on one occasion the princess's golden
```

Figure 2-4. Text and TeX control sequences in the Edit window.

All editing is done in the Edit window, and the standard Macintosh editing functions all work with this window. Textures also provides some additional editing functions; these are described fully in Chapter 3, “Editing,” beginning on page 76.

* The TeX command syntax is documented in *The TeXbook* as well as in other guides such as *A Beginner’s Book of TeX* and *TeX for the Impatient*—see page 224.

When you close a document's Edit window, the other windows attached to the document will close with it. You can print the contents of the Edit window whenever it is active (if the document has been typeset, you must turn off Flash mode to do this).

2.0 UPDATE: Command-clicking on a character in the Edit Window will now take you to the matching output text in the Typeset Window. See page 16.

Macros Menu Commands

`Macros` menu commands let you insert complex TeX commands in the Edit window in a single action. `Macros` menu items can range from simple formatting commands for selecting type styles such as bold and italic all the way to elaborate "mini-programs." You can create a file describing your menu with a simple pattern language, and paste it into a format's `Macros` menu to install it. The Macros menu is then immediately accessible to you through the Macros button (⊞) in the right sidebar of the Edit Window. For details, see Chapter 4, "The Macros Menu" (page 88). Some sample predefined menus can be found in the Samples folder—you can easily extend, modify, or replace these commands with your own menu items.

Options for Displaying Text in the Edit Window

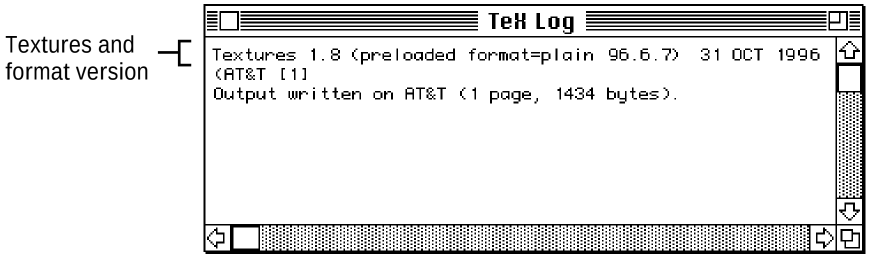
The `Edit` menu includes menu items for setting wrap and indent modes and for selecting a font and size for text in the Edit window. These `Edit` menu commands affect the appearance of the Edit window only and have no effect on typeset output. For details, see page 82.

Interacting with TeX: the TeX Log Window

The TeX Log window displays information and error messages from TeX about the processing of your text (Figure 2-5).

In Flash mode, the TeX log tracks TeX's actions but its window doesn't ever automatically open. When you select the `Typeset` command, the Log window automatically opens whenever TeX pauses, displaying error messages and letting you "talk to TEX" in order to interactively correct errors that TEX encounters.*

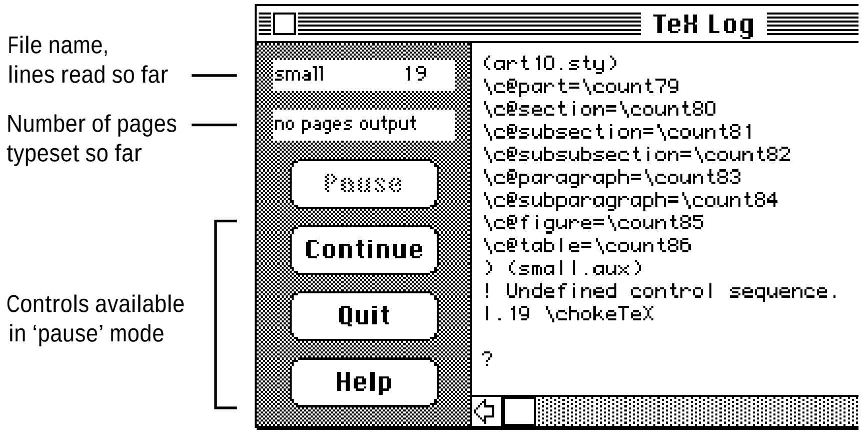
If, for example, you typeset one of the sample files and look at the TEX log, you'll see information about the format, the date, and—since there are no errors to report in our sample text—a final line telling you that the document has been typeset. In parentheses, you'll also see the number of pages and the size in bytes of your typeset document.



```
Textures 1.8 (preloaded format=plain 96.6.7)  31 OCT 1996
(AT&T [1]
Output written on AT&T (1 page, 1434 bytes).
```

Figure 2-5. The TEX Log window.

Some additional controls will appear on the left side of the Log window while typesetting is in progress; these change when an error occurs (Figure 2-6). In the shaded area at the top left of the Log window, Textures displays the name of the document and a running display of the line number that is being typeset on the current page. Below this, the number of each typeset page appears as it is completed. Four buttons—Pause, Continue, Quit, and Help—let you interact with the typesetting process.



small 19

no pages output

```
(art10.sty)
\c@part=\count79
\c@section=\count80
\c@subsection=\count81
\c@subsubsection=\count82
\c@paragraph=\count83
\c@subparagraph=\count84
\c@figure=\count85
\c@table=\count86
) (small.aux)
! Undefined control sequence.
l.19 \chokeTeX

?
```

Figure 2-6. The TEX Log window: error condition.

* The contents of the Log window are referred to as the "log file" in the TEX documentation.

Pausing TeX When the TeX Log window is active, you can click the **Pause** button to temporarily interrupt TeX. (If you're in Flash mode, this will abort typesetting.) TeX also pauses on its own when it encounters certain kinds of errors. When TeX pauses, the TeX log displays the line it was reading at the time and displays a question mark prompt (?) to indicate that it needs advice on how to continue. The three other buttons then become active:

Clicking the **Continue button** (or typing a carriage return) tells TeX to resume typesetting. (There are also options for telling TeX to run without stopping for errors, detailed on page 192.)

Clicking the **Quit button** (or typing `e`, for "edit") terminates the typesetting operation and returns you to the Edit window. Only the part of the document that has been typeset up to that point will then appear in the Typeset window.

Clicking the **Help button** (or typing `h`) tells TeX to write help information to the Log window.

TeX will remain paused until you click a button or type the keyboard equivalent. (The keyboard command must be followed by a carriage return.) While TeX is paused, you can also scroll back to view previous error messages.

Some common error messages such as `overfull hbox` won't stop TeX from continuing to typeset your text. (The results of such errors will appear in your Typeset window.)

Correcting Errors When TeX pauses for an error in "Typeset command mode," typing `e` or clicking the Quit button will return you to the Edit window at the point where the error occurred. You can then correct the error there and retypeset. You can also provisionally correct errors without terminating the typesetting operation by typing corrections *in* the TeX Log window itself. For more information on error handling and interacting with the TeX Log window, see page 191.

NOTE: There is one circumstance where you must type a correction *in the Log window* in order to terminate typesetting. If your document lacks an `\end` (or `\bye`) statement, you will get the error message

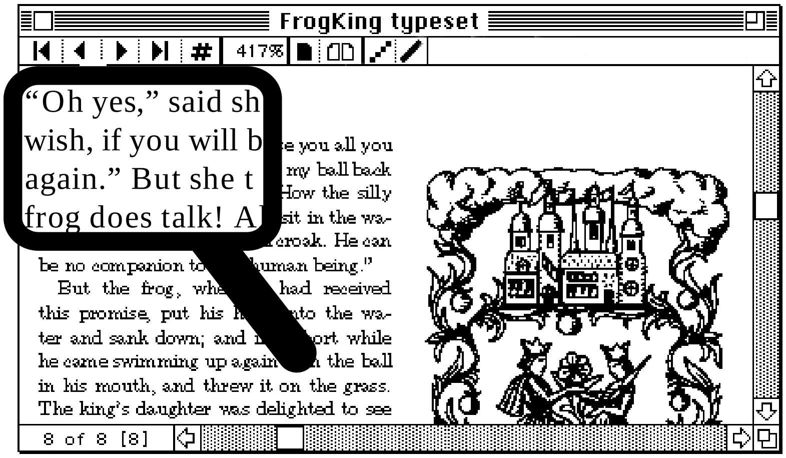
```
(Please type a command or say \end)
*
```

In this case, type `\end` to properly close the typesetting job. (This is not a problem in Flash mode, where typesetting terminates automatically without an `\end`.)

As we've mentioned, the TeX Log window is not attached to any particular document, and when you quit Textures, the contents of the Log window are not saved. But you can edit, save, or print the contents of the Log window in order to keep a record of error messages for correction at a later time. If you do save the TeX log, it will become an ordinary Textures text document.

Typeset Output: the Typeset Window

The Typeset window displays your typeset document as it will appear when printed (Figure 2-7). In Flash mode, the Typeset window remains in the background. When you typeset with the `Typeset` command, the Typeset window will come to the front after the first page has been typeset. You can use the various viewing tools to examine the typeset document or you can change the viewing magnification of the entire document, but you can't perform any editing operations or make changes to the contents of the Typeset window.



"Oh yes," said sh
wish, if you will b
again." But she t
frog does talk! A

e you all you
my ball back
How the silly
sit in the wa-
croak. He can
be no companion to human being."
But the frog, whe had received
this promise, put his h nto the wa-
ter and sank down; and i ort while
he came swimming up again n the ball
in his mouth, and threw it on the grass.
The king's daughter was delighted to see

Figure 2-7. The Typeset window.

You *can* Copy and Paste a page of the typeset document and process it as an ordinary Macintosh graphic, or save it as a separate file in either Adobe Illustrator format or in DVI format (TEX's binary device-independent format) by using the Save AsÉ command on the Typeset window. (These file formats are very different from Textures text files, and Textures can't re-edit such files after saving.) See "The Illustrator Interface" on page 72 and "DVI Files" on page 185 for details.

2.0 UPDATE: Command-clicking on a character in the Typeset Window will now take you to the matching output text in the Edit Window. See page 16.

Vertical Scrolling

Vertical scrolling through all of the pages of a document is automatically enabled whenever magnification is set to Fitto Window or greater. Scrolling allows continuous reading of each sucessive page of a document. You can go immediately to any page by moving the "thumb" of the scroll bar, while you are moving the thumb, the page number display indicates the page that will be displayed when you let go. When Two Page Display is selected, the page spreads will scroll accordingly.

View Controls Each Typeset Window contains a control bar with active indicators to select pages, control the viewing magnification, choose between single pages and spreads, and to enable the anti-aliased display.

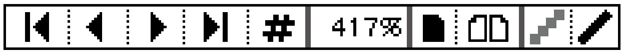


Figure 2-8. The View Controls menu bar.

These indicators let you choose between a single page display or a "two-up," side-by-side display that gives the appearance of a book format. Left and right pages are controlled by the TeX logical page numbers; even-numbered pages appear on the left, and odd-numbered pages appear on the right.

Choose between anti-aliased or ordinary black and white bitmap display. A bit of explanation: "aliasing" refers to the human eye's special sensitivity to the sharp edges and corners of the on-and-off pixels of the display; "anti-aliasing" is a sophisticated technique that uses intermediate (grey-scale) levels to reduce this effect. Anti-aliasing uses a grey-scale or color monitor to present a more accurate screen image of the document. While the screen will be more legible, it may appear "softer" due to the use of intermediate levels of gray.

Try anti-aliasing out for yourself. There is a tradeoff between the greater clarity of anti-aliased documents and the processing time that it requires – it will (unfortunately!) increase the time required for Textures to redraw the screen.

The magnification control in the View Toolbar shows the current display magnification. When pressed, a popup menu appears with a variety of magnification options available.

Paging through the Typeset Document

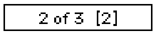
The page-select commands in the Tool Bar or View menu let you scroll through the typeset output page by page or go directly to a page you specify:

These controls allow you to step from the current page to the next or previous page, take you directly to the first or last page. These commands are also available as menu items under the View menu. Previous Page will appear dimmed when you are viewing page one. If Next Page is dimmed, you have reached the last page of your document.

#

Holding down this control or choosing `SelectPageÉ` from the View menu brings up a dialog so that you can go directly to any page in the document. You can type the number of the page you want to view, or use the scroll bar in the dialog to flip through the pages until you reach the one you want, and then click the “Go to Page” button. The page numbers are displayed as both physical and logical pages.

A page number in the selection dialog is usually a physical page number, but a decimal point can be used to indicate a logical (TEX) page number instead. Compound logical page numbers may be used (e.g., 6.3.1) if the document contains them. If two or more pages have the same logical page number, the first will be selected.


2 of 3 [2]

The page number display in the lower left corner of the typeset window shows the sequence number of the visible page (or page spread), the total number of pages in the document, and the logical page number. While scrolling vertically, there will often be portions of two pages or spreads visible; the page numbers displayed will be for the top page or spread.

An Aside: Page Numbers Textures uses two page numbering methods, both of which appear in the `SelectPageÉ` dialog:

- Physical (or sequential) page numbers count the number of typeset pages and always begin with the number 1.

- **Logical page numbers** are defined by the document structure. They have no necessary relationship to the physical page numbers, but usually have meaning to the document’s author. (For instance, negative logical page numbers might be used to refer to the front matter in a book.) The logical page numbers are always displayed in brackets, and appear in the typeset window’s title bar. A logical page number will sometimes be a compound number, for example to include a chapter or section number (see *The TEXbook*, p. 119).

NOTE: Both of these methods of page numbering may be different from the numbers that appear on your printed pages!

Page View Magnification The magnification control (417%) in the View Toolbar shows the current display magnification. When pressed, a popup menu lets you change the overall viewing magnification of your typeset document on the screen. (These menu commands don't affect your printed output.) You can select the following options:

- FitTo Window displays the entire page in the Typeset window, letting you check page layout and the placement of pictures.

- FitTo Width displays the page so that its entire width fits in the Typeset window, letting you examine the whole page by scrolling only vertically.

- ActualSize displays the typeset text at the size at which it will be printed.

- 120% Size. This is a good proofreading size.

- 200% Size is the Textures default size.

- 417% Size lets you examine the finer points of your typeset page. The 4.17 times magnification expresses the relation between the Macintosh screen and the LaserWriter printer (300/72 = 4.167), and shows what screen bitmap fonts will look like on a 300-dpi LaserWriter.

- SelectSizeÉ (M on your keyboard) lets you view the page at any size from 1/1000 ("1") to sixteen times its actual size ("16000"). The scroll bars in the Select Size dialog let you change the magnification in increments of ten, or you can type the size you want directly in the box. The Select Size dialog also displays a single character of ten-point type at the current magnification.

NOTE: The magnification scheme used in the Select Size... dialog is different from the "percentage" scheme used in the preset menu items listed above. In the Select Size... dialog, 1000 means 1.000, or 100%. Choosing magnifications of less than 1000 will display your document in the Typeset window smaller than it will be printed. (This numbering system corresponds to TEX's own magnification scheme.) The preset magnification options here are the same as those shown earlier: Actual Size (magnification 1000), ImageWriter (two times actual size, or 2000), LaserWriter (4.17 times the actual size, or 4167), and Fit to Window.

Visual Quality Some magnifications may look better or may be drawn more quickly than others because screen bitmap fonts at those sizes are available for screen display. Other magnifications may produce *scaled* bitmap fonts that display more slowly or appear jagged and coarse. This is not a problem for TrueType (outline) fonts, and the **Adobe Type Manager** can solve this problem for PostScript outline fonts as explained on page 131.

Cursor View Tools: Magnify, Measure, Scroll, and Zoom

When you position the pointer on a typeset page, the **magnifying glass cursor** will automatically appear. While holding the mouse button down, you can move the magnifying glass freely over the typeset document to examine kerning, ligatures, or other typographical details while keeping the overall view at a smaller size. Double click and hold the mouse button down to increase or decrease the area magnified.

The Option key by itself controls the ruler display within the magnifying glass. With the Option key down, click the mouse to display a tracking cross in the magnifier, and move the mouse to measure distances on the typeset page. (Release and press the Option key again to restart the measurement.)

Pressing the Command and Shift keys together ( ) gives the scrolling hand cursor; click the mouse and push the viewing area around within the typeset window. If you move off of the page while scrolling, the display will continue to auto-scroll towards the cursor.

Press the Shift key and click to magnify (zoom) the typeset display within the window. If you also press the Option key, the display will instead be reduced (shrink) with each click. (The cursor is the center of magnification or reduction.)

Printing

When Flash mode is off, the Print commands will print the contents of the currently active window (but not the Pictures window). If you print the Edit window, it will be printed in the form you see it on the screen, with TeX control sequences included. When Flash mode is on, the Print commands will normally print only the Typeset window, and you will need to close the Typeset window or turn off Flash mode in order to print the Edit window.

Use PrintOne to print just the current page from the active window. PrintOne starts printing immediately; no dialog appears and you can't choose any printing options.

NOTE: If you select the PrintÉ command while TeX is typesetting the document, Textures will not print the entire document, but only the pages that have been typeset up to that point.

When you print a Textures document from the Finder, the last typeset version of the document will be printed.

For more information on printing, see Chapter 10, "Printing," beginning on page 174.

Opening, Creating, and Saving Textures Documents

When you start Textures, an untitled window will appear automatically. You can also launch Textures with a number of documents open. From the Finder, click on any Textures document's icon, then shift-click to select any other files that you want to open at the same time. When you double-click on one of them, all will be opened at once.

Textures can read any **text-only** (or ASCII) file, and the OpenÉ command from Textures' File menu displays a list of all text-only files created by any application. A number of Macintosh editors such as the MPW (Macintosh Programmer's Workshop) Shell editor also produce text-only files. No special action is needed to import documents from these editors.

NOTE: When certain text editors save a Textures document, they may delete the typeset portion of the file along with any pictures that may be in the Pictures window.

Word-processing programs such as MacWrite and Microsoft Word have their own internal file structures that include formatting information such as font and style attributes. To be read by Textures, a document produced with one of these word processors must be saved as a text-only file. You can usually do this by selecting a “Text” or “Text Only” option under the application’s `Save AsÉ` command. A document saved in this manner will lose all formatting attributes.

NOTE: When saving a document as a text-only file in a word processing application, you may also have the choice of inserting “returns” as line breaks or to mark paragraphs only. If possible, you should select to insert returns as line breaks, not just after paragraphs. (Either one works for TeX, but without line breaks at the end of lines, you may see paragraphs as *very* long lines.)

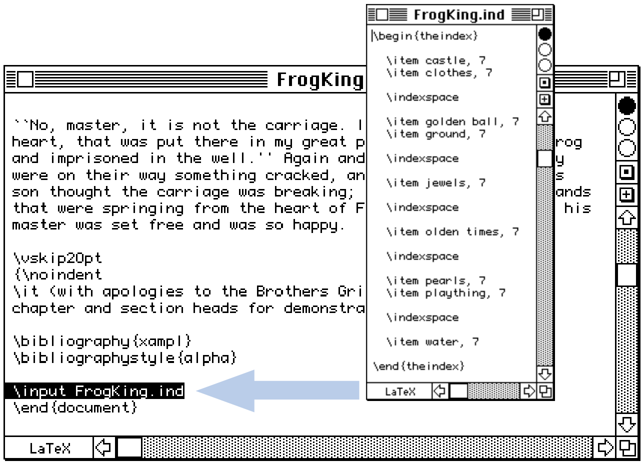
IMPORTANT! *Make sure that the files you import to TeX contain no control characters or other invisible characters (that is, other than carriage returns).*

Including Other Files in your Textures Documents

You can include other Textures text files in a document by using TeX’s `\input` statement. For example, the statement

```
\input option_keys
```

inputs a text file that contains a set of definitions for Macintosh Option-key characters. You can place your own definitions in input files to avoid having to individually include definitions in each file you typeset. (Alternatively, you can build your definitions into a canned format, for faster processing, as explained in Chapter 5, beginning on page 100.)



```
``No, master, it is not the carriage. I
heart, that was put there in my great p
and imprisoned in the well.'' Again and
were on their way something cracked, an
son thought the carriage was breaking;
that were springing from the heart of F
master was set free and was so happy.

\vskip20pt
{\noindent
\it (with apologies to the Brothers Gri
chapter and section heads for demonstra

\bibliography{xampl}
\bibliographystyle{alpha}

\input FrogKing.ind
\end{document}
```

```
\begin{theindex}

  \item castle, 7
  \item clothes, 7

  \indexspace

  \item golden ball, 7
  \item ground, 7

  \indexspace

  \item jewels, 7

  \indexspace

  \item olden times, 7

  \indexspace

  \item pearls, 7
  \item plaything, 7

  \indexspace

  \item water, 7

\end{theindex}
```

Figure 2-8. “Inputting” a text file. This example is drawn from the LATEX sample, “The Frog King.”

2.0 UPDATE: Project Master handles input files and master documents with great ease. See page 17.

Any files you plan to include should ordinarily be placed in the **TeX Inputs folder** or in the same folder as the document being typeset.

Where Textures Looks for an Included File

Textures looks for an included file in the following locations:

1 in the directory of the document that is being typeset;

2 in the TeX Inputs folder, and its sub-folders;

3 in the current directory (last opened by the user).

Macintosh filenames need not include extensions, such as ‘`.tex`’, but these may be present in the filename. If no extension is given for a TEX filename, the search will automatically look for both the filename as specified and with a ‘`.tex`’ extension appended.

When Textures is searching the TeX Inputs folder, a special rule applies: folders with names beginning with a bullet (‘¥’, option-8) will be searched *only* if the remainder of the name matches the current format name. This allows, e.g., a folder named ‘`¥LaTeX`’ to contain files that will be used only by the LATEX format.

These same search rules apply for any graphics files, such as EPSF or PICT files, that you are including in a document.

Alternatively, you can explicitly specify a partial or full **pathname** for your file so that Textures can find it in any location. For example, the pathname

```
\input HD:Textures:Macros:Article_macros
```

would always find the file "Article_macros" in the Macros folder in the Textures folder on the volume "HD."

IMPORTANT! *TeX itself does not understand spaces in file—or folder—names, and the names of input files cannot contain any spaces. (When TeX encounters a space, it stops reading the name as a filename.) You may give a Textures file a multi-word name for most other operations.*

NOTE: On Macintosh pathnames—because you could search in vain in your Macintosh user's documentation for any information on pathnames, we'll describe the rules here:

- Each directory (folder) name is delineated by a colon (:).
- A full pathname begins with the name of the device or root directory, as in the example shown above. A relative pathname begins with a colon, indicating the current directory.
- Two colons in a row denotes the parent directory, i.e., move up one level in the hierarchy.

Saving Documents

When you save the document in a Textures Edit window, the Typeset and Picture windows attached to it are automatically saved with it as well.* (To save a document, the Edit window must be the active window.)

The `Save` and `Revert` commands cannot be applied to Textures' Typeset or Picture windows alone. You can use the `Save AsÉ` command to save the Typeset window separately as an Illustrator or DVI file.

* Actually, they are added to the document's **resource file**, and so you don't need to explicitly save the file to save the latest version of the Typeset window or any pictures you've pasted.

Undoing Changes

The Undo command (Z) remembers all of the changes you have made to the text of your document since it was last saved, and allows you to undo each change, step-by-step, to the saved document. Redo (Z) steps forward reinserting each change made since the document was opened or last saved.

NOTE: Although rare, it is possible for the memory required for the Undo list to grow very large. If you are making many changes to a large document in a small partition, remember to save your work frequently.

The Revert item in the File menu removes any changes or additions you've made to the text of the document since you last saved it.

If you want to save the changes you have made and still preserve the original document, select Save AsÉ from the File menu, and enter a new file name.

Deleting the Typeset Portion of a Textures File

The typeset portion of a file requires considerable storage space and there may be circumstances when you want to delete it. You can use the following methods to delete only the typeset portion of your document, without affecting your basic text file or any pictures that may be in the Pictures window:

- With the Typeset window active, select Clear (B) from the Edit menu.

- Use the Delete Typeset utility available from Blue Sky Research.

NOTE: Opening the file with certain text editors and then saving it with that editor may also delete the typeset portion of your document along with any pictures that are stored in the document's Pictures window.

Document Context Information and Defaults

When you save a Textures document, a set of **"context" information** is saved along with it. The next time you open the document, you will again see it in the form that you saved it. Settings such as the following are saved with each document:

- Typesetting preferences (Flash mode on or off, format, e.g., Plain or LaTeX).

- Settings for the Edit window (window position, line wrap, font size, and so on).

- Settings for the Typeset window (size and location, viewing magnification).

- Settings in the Page Setup dialog.

You can also set your own **default settings** to be applied to any new document. Simply prepare a document the way you want it, and save that document with the name "**Defaults**" in the **TeX Formats** folder. (To save settings for the Typeset window such as viewing magnification or position, you should also include some text in the Defaults document and typeset it; that text will not appear in new documents.) After you restart Textures, Textures will then use these default settings each time you create a new document.

This means, for example, that if you commonly turn wrap mode on, Flash mode off, position the Edit window on your small screen and the Typeset window on your larger color screen, use the LaTeX format, and like to view pages at 110% size, then you can create a "Defaults" document with these characteristics and place it in the TeX Formats folder. Thereafter, new documents you create will have those same settings and window positions.

Exporting Typeset Material to Other Applications

You can use two methods to transfer the typeset page of a Textures file to other Macintosh applications:

- Copy and paste. When the Typeset window is active, the `Edit` menu's `Copy` command will copy the current typeset page to the Macintosh Clipboard.

- Save the document as an Adobe Illustrator document. With the Typeset window active, “Illustrator 88” will appear as the default file format under the File menu’s Save AsÉ command.

There is a difference between these two methods: if you copy a page with the Copy command, you will not get the full precision available with the Save AsÉ “Illustrator 88” option.

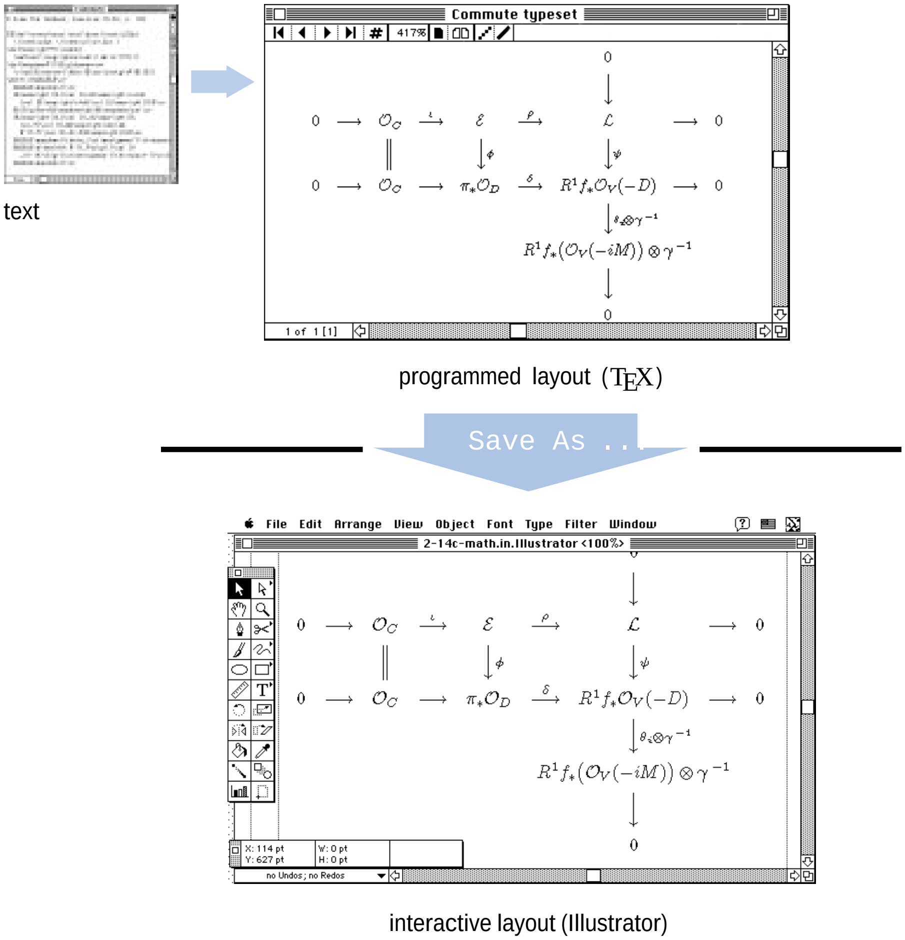
Copying a Typeset Page to Another Application

You can copy an entire typeset page to the Clipboard with the Edit menu’s Copy command; you can then Paste it into the Scrapbook, or into any application that accepts picture material (including to Textures’ Pictures window). However, the Clipboard export facility is of limited precision, so that character kerns, rule sizes and positions, and other graphic relations may be distorted; for better results, we recommend the Adobe Illustrator interface.

The Illustrator Interface

Textures’ Illustrator interface makes possible a new way of looking at TeX, as a typographic input to a graphic-arts process. In the usual TeX paradigm, TeX forms the end of the typesetting process: an input text file is passed to TeX and TeX produces a typeset output file, which it is then up to your particular installation to print. In the paradigmatically shifted Macintosh system, TeX can produce input to another process, as illustrated in Figure 2-9.

The “Illustrator” interface is also an EPSF interface: that is, the “Illustrator” file you create can be read and used by many applications other than Adobe Illustrator, such as Aldus PageMaker, Quark XPress, and Microsoft Word. (Textures’ Save AsÉ Illustrator 88 option can be advantageous because Illustrator 88 is a much “leaner” program than Illustrator 5.0, requiring considerably less memory to run.)



$$\begin{tikzcd}
 & & & 0 \arrow[d] & \\
0 \arrow[r] & \mathcal{O}_C \arrow[r, "\iota"] \arrow[d, equal] & \mathcal{E} \arrow[r, "\rho"] \arrow[d, "\phi"] & \mathcal{L} \arrow[r] \arrow[d, "\psi"] & 0 \\
0 \arrow[r] & \mathcal{O}_C \arrow[r] & \pi_*\mathcal{O}_D \arrow[r, "\delta"] & R^1f_*\mathcal{O}_V(-D) \arrow[r] \arrow[d, "\theta_*\otimes\gamma^{-1}"] & 0 \\
 & & & R^1f_*(\mathcal{O}_V(-iM))\otimes\gamma^{-1} \arrow[d] & \\
 & & & 0 &
\end{tikzcd}$$

Figure 2-9. Textures' interface to Illustrator.

Textures Illustrator interface means that the highest quality typography can serve as digital input to an industry-standard graphic arts program, letting you produce complex, one-of-a-kind layouts, add color, or merely add fine details or nudges. (And we do mean one-of-a-kind: the difficulty with crossing the threshold from TEX's programmability into the hand-manipulated "WYSIWYG" world is that you can't go back gracefully—so you need to set up your typography entirely to your satisfaction before transferring the file to Illustrator.) Copying a typeset page to Illustrator may be useful for something as straightforward as making a graphic of an equation or other typeset image and adding color to it. Other examples of things that you can easily do with the Illustrator interface that are not easily done in TEX itself are rotation, adding additional elements, cutting and layout.

Once in Illustrator, you can select parts of the typeset page as individual objects. (Object boundaries fall at kern points: each word is selectable as an individual object, and there may be break-points within words themselves.)

NOTE: An Illustrator-format file is a type of PostScript file—specifically, a very limited subset of PostScript operators understood by Illustrator. This format may be less efficient for "conventional" TEX jobs because each character is a graphic object.

This concludes our survey of Textures basic operations. The following chapters will look in turn at the specifics of editing, macros menus, the use of canned formats, and fonts.

chapter 3

Editing and the Edit Window

The Edit window is your “working window” onto a Textures document, the place where you input text, make editing changes, and write TEX commands. Textures supports the standard Macintosh editing techniques as well as additional commands and options for displaying text in the Edit window. You can also customize your editing environment and tailor it for each format that you use by defining your own macros as menu commands. In this chapter, we briefly survey Textures’ standard editing functions. We’ll look at building your own Macros menus in Chapter 4.

CONTENTS

Standard Editing Functions

The Textures editor includes all of the standard Macintosh editing techniques for inserting text, selecting, replacing, deleting, cutting, copying, and pasting text. All of the keyboard command equivalents also operate here as they do in any standard Macintosh application. If you aren't familiar with Macintosh editing functions, please refer to your *Macintosh User's Guide*. The Textures editor also includes a set of search-and-replace functions and supports all of the standard "arrow-key" functions, which are summarized below.

Editing with the Arrow Keys

You can use the arrow keys together with the Command ( ) and Option ( ) keys to *move the insertion point* as follows:

beginning of line

end of line

top of window (page up when at top of window)

bottom of window (page down when at bottom of window)

beginning of previous word left

end of next word right

In combination with the Shift ( ) key, you can *select text and extend the selection* as follows:

extend selection one character to the left

extend selection one character to the right

extend selection by one line up

extend selection by one line down

select/extend selection one word left

select/extend selection one word right

select text from insertion point to start of line,
extend selection one line up

select text from insertion point to end of line,
extend selection one line down

select text from insertion point to
top of window,
extend selection one screen up

select text from insertion point to
bottom of window,
extend selection one screen down

Brace matching

When you type a left or right brace (‘`{`’ or ‘`}`’) the editor will briefly highlight, or flash, the corresponding brace that starts or finishes the group being formed. This helps you tell just which brace you’ve matched, and how many braces will close the group you’ve finished.

Note that it’s easy to fool the brace matching with complex TEX constructions, but this isn’t usually a problem.

Using Macintosh Option-key Characters

To use Macintosh Option-key characters in your TEX files, include the command

```
\input option_keys
```

at the beginning of your Textures document. This command will include the file **option_keys** (in the TeX Inputs folder); this file contains a list of available Option-key characters and their TEX definitions. You can change these definitions to meet your needs.

IMPORTANT! Files containing Macintosh option-key characters will not be portable to other TEX systems. If you plan to transfer your *files you will need to use standard TEX methods for inputting special characters, as documented in* The TEXbook*.*

Multiple Undo and Redo

The Undo (Z) command in the File menu lets you undo each change that you have made in the Edit Window back to the point when the file was opened or last saved. Redo (Z) reinserts changes removed by the Undo command. Together these commands let you move back (and forward) step-by-step through changes made to a document. Once you save the document, the Undo list is cleared, and a new list begins with additional changes.

NOTE: Although rare, it is possible for the memory required for the Undo list to grow very large. If you are making many changes to a large document in a small partition, remember to save your work frequently.

Moving Between the Edit and Typeset Windows

2.0 UPDATE: It is now possible to move easily between text in the Edit Window and items in the Typeset Window. See page 16.

Finding and Changing Text

The Edit menu's Find and Change commands let you locate and change text in the Edit window. (Find also works in the Log window.) Any text that is already selected in the Edit window will initially appear in the "Search for" line of the FindÉ and ChangeÉ dialogs.

- FindÉ (F): locate any text string in the Edit window and highlight the first instance of the string. (This search string can be any sequence of characters including characters in TEX control sequences.) If the search reaches the end of the document without finding the string, you'll hear a beep.

- Find Same (G): select the next occurrence of the last search string you specified.

- ChangeÉ (H): find and replace a specified text string. Clicking the “Change” button will change the first occurrence of the search string. “Change All” will replace every instance of that string from the current position of your insertion point to the end of the file. (Select the “Wrap Around” option to change every instance in the entire file.)

- Change Same (J) repeats the most recent ChangeÉ operation.

The Find and Change commands begin to search from the *current position* of the insertion point in the Edit window and move down the page to the end of the file. You can cause the search to wrap back to the beginning of the file by selecting the “Wrap Around” option.

Note that the Change commands will change the next occurrence of the search string *before* you can see what and where it is. The Undo command lets you undo the most recent change you have made. (Undo will also undo a Change All.)

Reverse Search Hold the Shift key down while selecting Find or Change, and the search will scan backwards through the document. (There’s also a check box in the search dialogs.)

Wildcard Search Textures 1.8 defines two special characters in search strings: the ellipsis (‘É’, option-semicolon), and the left quote (‘Ô’, option-right-bracket). The ellipsis (‘É’) is a wildcard character, matching any string of characters. It may appear only once in the search string, and it may not appear as the first or last character. A single ellipsis may also appear anywhere in a *replacement* string, if one appears in the search string; in each replacement, the ellipsis will be replaced by the corresponding source text.

The left quote (‘Ô’) character quotes the next character in the search or replacement string, i.e., the next character is taken literally even if it’s one of the special characters. (You’ll need this only if you need to search for one of the special characters.)

The space character is also treated specially in Textures 1.8 searches: it now matches any “white space”, i.e., any number of consecutive spaces, tabs, or carriage returns. This means, for example, that you can search for “two words”, and you’ll find text containing those two words even if they’re separated by a line break. (To search for a literal single space, quote the space character, as above.)

Finding Invisible Characters In the Find and Change dialogs, there is no way to specify invisible characters such as carriage returns or linefeeds *per se*. However, if a piece of text has been selected in the Edit window, this text will initially appear as the search string in the `FindÉ` and `ChangeÉ` dialogs. You can use this feature to search for invisible characters such as carriage returns: first select the carriage return (or search string text that includes a carriage return), and then select the `FindÉ` or `ChangeÉ` command. You can also directly copy and paste text (C , V) into the Find and Change dialogs. You can thus copy and paste a carriage return or other invisible character from the Edit window into the Find and Change dialogs.

Going to a Specified Location in the File

The Marks control ⊡ in the Edit window sidebar and `Line #É` commands let you go directly to a specific location in the file:

Marks invisibly marks any text selection or location in the file with a name you specify. You can then return to that point in the file by choosing that name from the Marks menu. Place the cursor or select a piece of text, choose ⊡ `Add MarkÉ`, and assign a name to it. To remove a mark, choose ⊡ `Unmark` and the name of the mark you want to remove.

`Line #É` (L) brings up a dialog asking for the line number you want to locate: type the line number, then click “OK,” and the window will scroll to that line. This command is useful for finding and correcting errors that are identified by line number in the TEX Log window.

Edit Window Display Options

To let you alter the display of text in the Edit window for maximum convenience and readability, the `Edit` menu also includes the commands `Block Indent`, `Indent`, `Wrap`, and `FontÉ`. These commands have no effect on typeset output.

The Edit Window Display Font

By default, your input text appears in 9-point Monaco type. You can change the display font and size by selecting the Edit menu's Font command. The display font in your Edit window does not affect your typeset output.

(Note that the Font command's list of screen fonts is different from the File menu's Show FontsÉ list of font metrics. These are screen bitmap fonts and not the fonts used to produce high-quality typeset output.)

Line Lengths and Paragraphs ("Wrap")

When you typeset a document, TeX automatically determines where to optimally break the lines of text for the column width and justification that you have specified. Therefore, when you enter text in the Edit window, you can freely break a line of text at any point: TeX interprets a single carriage return as a space and thus takes no notice of the line lengths of input text. To begin a new paragraph, type *two* carriage returns (that is, create a blank line).

For your own convenience, you will likely want to keep the text that you type fully visible in the Edit window. To make this happen automatically, select Wrap from the Edit menu. When Wrap is selected, a check mark appears next to the menu item, and Textures will confine all text lines that you subsequently type to the current width of the Edit window. (Textures does this by inserting carriage-return characters at the end of each line.)

Wrap remains active until you return to the Edit menu and de-select it. With Wrap off, line breaks will occur only when you actually type a carriage return. (The wrap on/off setting is part of the context information saved with each file.) Whether Wrap is selected or not, you can always break lines of text manually by typing a carriage return.

To wrap text that has already been entered, select the text and use Wrap Selection (E).

Wrapping Text but not TeX Commands

Since carriage returns are reinterpreted as spaces, adjacent lines may be consolidated onto a single line if you rewrap an entire document. We prefer to automatically wrap text but not to wrap TeX commands, which are often sensitive to line structure; thus, wrap mode also includes the following features:

- Any *initial space* will cause wrapping to be avoided for that line: line wrapping will affect only lines that begin at the left margin.

- The `\`, `%` , or `$` characters at the beginning of a line will likewise prevent the line from being wrapped into the previous line.*

More specifically, the *beginning* of the line will not be wrapped to the preceding line, while the end of the line will continue to wrap. So, simply start each code or math line with a space, `\`, `%`, or `$`, and wrapping will "avoid" that line. (TeX itself is usually not sensitive to initial spaces in text.) Setting Indent in the Edit menu will automatically maintain an indentation level (i.e., will insert spaces) as you type.

NOTE: If a TeX command is embedded in a line of text and that line is wrapped to the end of a comment line (a line beginning with the % character), the result will be to "comment out" part or all of your TeX command, generating TeX errors when you typeset.

Indenting Text in the Edit Window

When the Indent mode is on, any new line you type is automatically indented to match the line immediately above it. You'll probably use Indent primarily for entering TeX commands. To indent lines of text:

1. Manually indent the first line that you wish to indent by typing the desired number of spaces at the beginning of the line.

2. Select the Indent menu item to continue to use this indentation for subsequent lines of text.

A check mark will appear next to the Indent menu item, which will remain active until you return to the Edit menu and turn off indenting. You can't use Indent to go back and indent text that has already been entered.

* This set of characters can be changed, if desired, by editing the STR# 259 resource with ResEdit.

Using Block Indent

To indent a selection of text which is already entered in the Edit Window, use the `Block Indent` command:

1. Highlight the text you wish to indent.

2. Select the `Block Indent` menu item and release to move the highlighted text one space to the right. The command may be repeated to move the selection further to the right.

3. Hold the shift key ( ) down to change the menu item to `Block unindent`. This command will move highlighted text one space to the left.

Using Block Comment

If a line of text or a TEX command is preceded by the % character, it is “commented out”, and TEX will ignore any text to the right of the % character when typesetting. In certain cases, it may be helpful to automatically “comment out” whole paragraphs or large sections of a document. To block comment multiple lines of text:

1. Highlight the text you wish to “comment out”.

2. Select the `Block Comment` menu item to place a % character as a prefix to each line.

3. Hold the shift key ( ) down to change the menu item to `Block uncomment`. This command will remove % characters that are the first characters in the selected text lines.

A Note: Using other Text Editors with Textures

As we’ve mentioned, Textures will process text-only files created by any application, and there may be circumstances when you want to edit your files with an editor other than the built-in Textures editor. For example, the Macintosh Programmer’s Workshop (MPW) Shell editor is a powerful, “UNIX-style” scriptable editor that you may find very useful in automating repetitive tasks including complex search-and-replace operations on a large number of files. However, it is *not* a good idea to have the same file open simultaneously in Textures and in another application—edit and save your files in one application before opening them in another.

You can, however, drive Textures remotely via AppleEvents from external, programmable editors. Examples are the Alpha editor (available from `cs.rice.edu`) or Emacs for the Macintosh, available from `ftp.cs.cornell.edu`. See the Blue Sky Research FTP server for sample Alpha scripts.

We'll now look at the use of the `Macros` *menu and canned formats, two key elements that work together to lend both power and ease of use to the Textures system.*

chapter 4

The Macros Menu

User-defined macros as menu items streamline the process of entering TEX commands and editing text in your Textures documents. Not only is the Macros menu completely customizable, but you can customize a Macros menu for each format: each Macros menu is associated with a specific format so that when you switch from Plain to LATEX, for instance, the contents of the Macros menu change as well.

In this chapter, we'll look at how to install macros in the Macros menu, how to write your own menu macros, and along the way, we'll give some examples of useful possibilities. The file "Macros Menu Samples" in the Samples folder contains the text of the Macros menu items that are referred to in this chapter.

CONTENTS

User-Defined Macros as Menu Commands

The `Macros` menu greatly speeds typing and editing by giving you direct, convenient access to your most frequently used TeX commands. These commands can be simple or elaborate macros (or just commonly entered strings of text), enabling you to input TeX commands with a single keystroke. Menu macros are an aid to organization and memory, providing a kind of on-line reference to your TeX macros and letting you "automate" a consistent programming style in your TeX commands. By the same token, if you are new to TeX, macro menus will help you to avoid syntax errors and learn TeX more quickly.

The `Macros` menu is programmable, so the menu structure and actions are completely under your control.

Formats and the Macros Menu

You can define a specialized `Macros` menu for each format listed in the `Typeset` menu, so that you can "package" your TeX programming together with an easily customized user interface. There is actually a separate menu (and menu program) for each format: the current `Macros` menu changes automatically as you switch documents and formats. For speed and ease of use, we highly recommend that you create a separate format, and `Macros` menu, for each of your different Textures functions. (For a full discussion of formats, see Chapter 5, beginning on page 100.)

Building your own Macros Menus

A `Macros` menu is defined by a simple program that you copy from an ordinary Textures file, with one line for each menu item. As Figure 4-1 shows, the process of making a new menu is accomplished in two stages:

1. Create or edit your menu text in a standard Textures file.

2. `Copy` (C) the menu's text and paste it with the `Macros` `Paste Program` item.

You can do the reverse as well: copy an already installed menu with the `Copy Program` item, and then paste it (P) to a text file (or use `Paste Program` to attach it to another format).

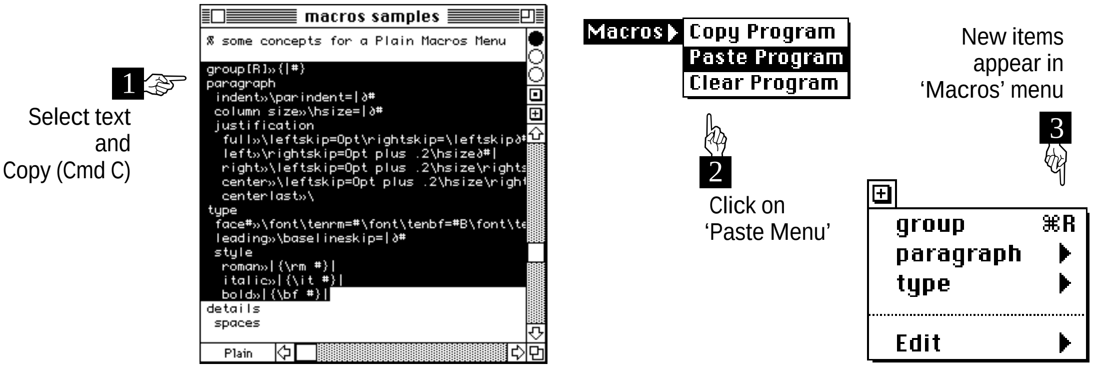
Clearing the Macros Menu for a Selected Format

To remove a Macros menu attached to a specific format:

1. Under the Typeset menu, select the format of the Macros Menu you wish to remove.

2. Select Macros Clear Program item.

3. The (⊞) in the Edit Window sidebar will appear grayed-out until a new Macro Menu is programmed for the format.



```
% some concepts for a Plain Macros Menu
group[R]»{|#}
paragraph
 indent»\parindent=|∂#
 column size»\hsize=|∂#
 justification
  full»\leftskip=0pt\rightskip=\leftskip∂#
  left»\rightskip=0pt plus .2\hsize∂#|
  right»\leftskip=0pt plus .2\hsize\rights
  center»\leftskip=0pt plus .2\hsize\right
  centerlast»\
type
 face#»\font\tenrm=#\font\tenbf=#B\font\te
 leading»\baselineskip=|∂#
 style
  roman»|{\rm #}|
  italic»|{\it #}|
  bold»|{\bf #}|
details
 spaces
```

Select text and Copy (Cmd C)

Click on 'Paste Menu'

New items appear in 'Macros' menu

Figure 4-1. Making a new Macros menu.

Ordinarily, you'll maintain a working document in which you define and revise your menu programs. You may repeatedly use the same menu items (including Command-key equivalents) in any Macros menu where it's appropriate.

Your Textures package includes sample macro menus for the Plain and LATEX formats. We'll now look at how to write your own macros menus by reference to these sample menus, so you may want to go ahead and install a sample menu at this point.

Installing a Sample Menu

The file "Macros Menu Samples" contains several sample Macros menus. You can install one of these sample menus as follows:

1. Under the Typeset menu, select the format to which you will attach the sample menu (e.g., Plain).

2. Open the “Macros Menu Samples” file, *select the text for an entire menu*, and copy it with the `Edit` menu’s `Copy` command (`C`). (You install a `Macros` menu entire, not one menu item at a time.)

   NOTE: Don’t use the `Macros` `Copy Program` item; `Copy Program` copies the currently installed menu to the clipboard.

3. Select the `Macros` `Paste Program` item (from the `Macros` menu) to install the new menu definitions.

4. To remove an installed Macro Menu, Select `Macros` `Clear Program` item.

You may now close the “Macros Menu Samples” file. Your new menu is installed and ready for business. If you plan to use more than one format, repeat the same procedure with a sample `Macros` menu for the other format(s).

NOTE: The sample menus we’ve provided merely demonstrate some techniques, and we don’t necessarily recommend these menus as best suited to your particular needs or working style—they are freely customizable, a starting point only.

Writing your own Menu Macros

You define `Macros` menu items by means of a menu “programming language” that consists of exactly ten special characters. It is so simple that it is best learned by example, so at this point we recommend that you read and experiment with the “Macros Menu Samples” document provided with your Textures package. These special characters are summarized in Table 4-1 and explained thereafter.

Symbol	Keystrokes	Meaning
¨	\	Define a menu item (`item-name ¨ item-action`).
#	3	Represents any text that is currently selected in the Edit window.
|	\	Place the cursor (*two* | 's indicate a selection range).
¶	d	Insert a carriage return.
-	-	Menu divider.
@	2	Copy `item-name` into result.
[*key*]	[and]	Define a Command-key ( ) shortcut.
Ô	]	Quote one of the above special characters for literal inclusion. (*Not* the left quote/tilde key.)
%	5	Comment out the rest of the line.

Table 4-1. Macros menu special characters, with the Shift [ ] and Option [ ] keys.

We'll now look at the use of each of these characters in writing a Macros menu.

Structure of a Menu Item

There is one line per item. (A menu item cannot be wrapped in the Edit window—turn off Wrap when you write your menu macros.) The basic format is

```
item-name ¨ item-action
```

where `item-name` is the name that will appear as a menu item in your Macros menu (⊞), ¨ is the "replace character," and `item-action` is the menu action. At its simplest, `item-action` could simply be text, in which case it will simply be pasted to the Edit window, replacing any currently selected text or being inserted at the insertion point.* Menu actions are limited to 255 characters each, and each menu item must be written as a single line.

* If you omit `item-action` entirely, the name of the menu item will appear in the menu, but with no action specified. If the menu action is thus empty, executing the menu item will delete any currently selected text in the Edit window; but if no text is selected, nothing happens.

The “replace character” (¨) divides the line into two parts.† The left part is the name of the menu item, and the right part defines the menu action (the macro) itself. This macro text will (1) replace the currently selected text, if any, or (2) will be inserted at the current selection point. (If no menu action is specified, any current selection will simply be deleted). For example,

```
Setup¨\hsize=20pc¶\parindent=0em¶\parskip=1em¶
```

produces a menu item

which inserts into your document a sequence of TEX commands, separated by carriage returns:

```
\hsize=20pc
\parindent=0em
\parskip=1em
```

Special Characters

We’ll now look at the special characters in detail. (The Key Caps desk accessory may be useful in remembering these keys.)

Current Selection

The # character (3) lets you select text and position a command around it. For example, the menu item

```
Boldface ¨ {\bf #}|
```

will apply Plain TEX’s `\bf` command to any selected text: when you choose the menu item, the # is replaced by the current selection (if any).

The # character can appear more than once; for example

```
Typeface ¨ \font\tenrm=#\font\tenbf=#B\font\tenit=#I\rm¶
```

will define a selected font name to be the font family for the Plain TEX default (i.e., instead of the CM fonts).

† This is the right “guillemot” (French quote). Because we don’t know how to pronounce *guillemot*, we’ll call this a “chevron.” On an American keyboard, you get it by typing Shift-Option-Backslash.

| **Insertion Point (or Selection)** A single | (\) sets the new location for the *insertion point* on selection of the menu item. Two | 's set the beginning and end of the *text to be selected* on completion of the menu action. For example, in the "Boldface" example given above, the cursor will be placed immediately after the affected text.

You can also write a macro to provide a framework for text that you will subsequently enter, with the cursor positioned in the proper place for you to start typing. For example, we could extend our "Setup" example as follows:

```
Setup¨\hsize=20pc¶\parindent=0em¶\parskip=1em¶¶|¶\bye
```

For the following result, with the cursor positioned for entry:

```
\hsize=20pc
\parindent=0em
\parskip=1em



\bye
```

(A LaTeX example is given on page 98.)

¶ **Carriage Return** The ¶ character (d) inserts an explicit new line in the macro text.

@ **Copy Name** The @ character copies the name of the macro into the macro text. This is especially useful if you're making a menu that's simply a list of TeX macro names.

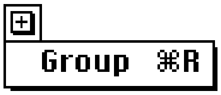
Quote The left quote ( ]) lets you include the special characters used to program the Macros menu (¨, #, | , ¶, @, Ô, %, [,]) as literal characters in the macro text. For example, Ô# gives a literal sharp sign in the resulting text.

% **Comment** The % character (5) ignores the rest of the line.

Command-Key Shortcuts To assign a Command-key equivalent to a menu item, put the key character in brackets immediately after the item's item-name. For example,

```
Group[R] ¨ {|#}
```

will create a new item, `Group`, with the Command-key equivalent R .



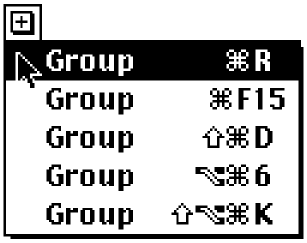
(This menu item, `Group`, will simply enclose the current selection in braces and place the cursor at the beginning of the group.)

The key characters available for use include the function keys (F5 - F15), and the following alphabet keys: `[D] [I] [R] [K] [Y] [U] [4] [5] [6] [7] [8] [9]`.

The shift key and the Option key may be used in combination with the Command key to extend the set of Command-key shortcuts. Use the '!' symbol to represent the Shift key, and the '@' symbol for the Option key. The following example shows the same macro assigned to a range of possible key combinations:

```
Group[R] ¨ {|#}
Group[F15] ¨ {|#}
Group[!D] ¨ {|#}
Group[@6] ¨ {|#}
Group[!@K]  ¨ {|#}}
```

which creates a Macro Menu with various Command-key combinations displayed for the `Group` macro.



Hierarchical Menus You can create hierarchical (multiple-level) `Macros` menus, with sub-menus for major topics, simply by indenting the menu program lines. To start a new level or sub-menu, move to the right *one space* when you start the next line, according to the following rules:

- All main menu items must begin *at the left edge* of the window.

- Omit the right side (i.e., the menu action) of a “parent”-level menu.

- Sub-menu items are indented one space from the left edge and are placed on the next line below the main menu item.

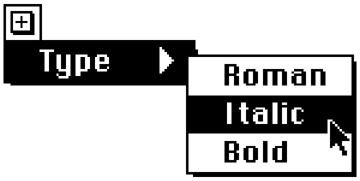
- Second level sub-menu items are indented two spaces, and so on. Up to five levels of hierarchy are permitted.

- To end a level, move back to the left one space.

For example, the program text

```
Type
 Roman¨{\rm #}|
 Italic¨{\it #}|
 Bold¨{\bf #}|
```

will yield the menu

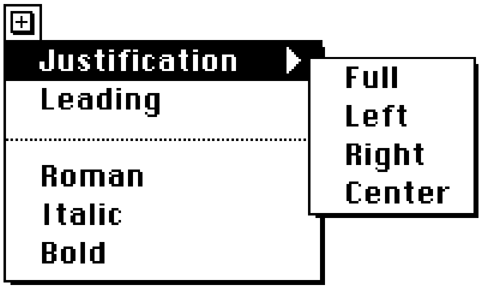


(Of course, in actual practice it is most convenient to place your most frequently used menu items at the top level for rapid selection and to subordinate items that have a lower frequency of use.)

Menu Dividers A dash or hyphen on a separate line creates a divider in the menu, (i.e., the gray line separating sections). Dividers have no action, and cannot be selected; they help to visually organize complex menus. For example, the menu program

```
  Justification
   Full¨\leftskip=0pt\rightskip=0pt¶#|
   Left¨\leftskip=0pt\rightskip=0pt plus .2\hsize¶#|
   Right¨\leftskip=0pt plus .2\hsize\rightskip=0pt¶#|
   Center¨\leftskip=0pt plus
.2\hsize\rightskip=\leftskip¶#|
  Leading¨\baselineskip=|¶#
  -
  Roman¨{\rm #}|
  Italic¨{\it #}|
  Bold¨{\bf #}|
```

creates the menu



Sample LATEX Menu

To use one of the sample LATEX menus, select the `LaTeX` format and install the menu as explained above on page 91. Then start building your document. (If you are unfamiliar with LATEX, you may want to read the next chapter at this point and then return to this section.)

1 From the `Macros` menu, select the `documentclass` and point size you prefer. When you release the mouse button, the correct code will be inserted into your document.

2 Next create the shell of your document, choosing the bracketed `begin` and `end` document statements. The cursor will now be blinking in the correct position for you to start entering text inside of this frame.

The macros for LATEX have been written so that you simply select some text to be placed in an environment and then select that environment: the macro will be pasted around your selected text, and you can move on to your next operation. We've provided environments to choose from, `fontsize` commands, and various math symbols, to make creating LATEX documents quick and easy. (Refer to your LATEX documentation for more details.)

We'll now proceed to look at the specifics of using LATEX and other TEX formats with Textures.

chapter 5

Reinventing TEX: Macros and Formats

We've emphasized that TEX is a *programmable* typesetting system. This means that you can use TEX's basic commands to define your own TEX commands, or **macros**. You can, in turn, combine sets of macros into custom **formats**, giving you the ability to create your own typesetting "language" for any general or specific purpose. And we've already examined how Textures links your formats to user-definable macros menus, enabling you to customize each format's user interface for maximum convenience and efficiency.

This chapter explains the use of pre-compiled formats with Textures. It also introduces LATEX, a popular, more highly structured variant of TEX, along with some other standard formats such as AMS-TEX.

Finally, we'll look at how to turn your own macros into custom formats.

CONTENTS

TEX as an Extensible System

As we mentioned in the Introduction, TEX is built around a central programmable core, to which various “layers” are added. This rudimentary TEX and the Plain TEX format that is built on top of it form a larger core upon which further specific “applications” can be built.

In a very literal sense, each TEX file is a *program:* in the more traditional TEX way of doing things, you execute this program when you select the `Typeset` command; in `Flash mode`, program creation and execution become a single process. (So, if you are a TEX beginner who has progressed thus far, Congratulations!—you already are a TEX programmer.)

To package up the programming of your specific applications, TEX provides various tools, including:

- `\def` **statements** to define your own macros. You can then use these macros exactly as you would use built-in TEX commands.

- **input text files**, which may be any combination of text and TEX commands (see page 67).

- **formats**: “canned” sets of TEX commands (which may include your own macros) that TEX reads in before it starts processing a file. These serve to extend standard TEX.

In general, your TEX files will begin with a sort of “preamble” consisting of page and paragraph parameters (`\hsize`, `\baselineskip`, `\parskip`, etc.), `\font` statements defining which fonts your document will use, `\def` statements defining your macros, and so on (Figure 5-1). This is exactly the sort of information that you can conveniently define as a canned format.

```
  % Dimensions
  \hsize 24pc             % column size 24 picas
  \baselineskip 12pt      % set type 10 on 12 points
  \parindent .5in         % half-inch paragraph indent
  \parskip 0pt            % no space between
paragraphs
  % Define Fonts
  \font\tenrm = Times at 10pt    % standard roman font
  \font\tenbf = TimesB at 10pt   % bold family
  \font\tenit = TimesI at 10pt   % and italic
  % Select a font
  \rm                            % or \bf or \it
```

Figure 5-1. Elementary preamble to a TEX file.

To these basic TEX capabilities, Textures adds the ability to customize your user interface as well, by adding macros as menu items in the Macros menu. Each Macros menu that you define is connnected to a specific format, so you can have as many different Macros menus as you have different formats.

Defining your own TEX Macros

TEX's `\def` statement lets you define your own macros for subsequent use. A simple example might be:

```
\def\page{\vfill\break}
```

This defines a macro, `\page`, which consists of two TEX commands, `\vfill` and `\break` (that is, it executes a page break). Once defined, the new macro can be used in your files just like any other TEX command. For a full explanation of `\def`, refer to your TEX documentation.

Numerous macros are available in the public domain, and you may want to explore what is out there before reinventing the wheel.

The CD Extras folder of the Textures CD-ROM contains all of the macro packages available on CTAN at the time the CD was burned. For an extensive survey of what is currently available, the TEX-Index is the best source (see page 222).

About Formats

As we've seen, a TEX **format** is a packaged set of TEX commands that you can use to define a particular document style and layout. The range of what we mean by document style and layout should not be thought of too narrowly: a format can define anything from a book style, to macros for address labels, for database publishing, for producing bar codes, calendars, and on and on. Textures lets you compile such a set of TEX commands into a precompiled or "canned" format file. To install a canned format, you simply place it in the **TeX Formats folder**; its name will then appear in your `Typeset` menu and you can use it to typeset your documents. A TEX format functions as an extension of standard TEX, meaning that if you define a new command in your format file, you can then use that command anywhere in your document just as you would use one of the built-in TEX commands.

For the sake of speed, formats are contained in binary files, but they could also be stored in ordinary text-only files. (That is, adding a format on top of Plain TEX is logically identical to reading in a text file containing the same definitions by using an `\input` statement at the beginning of your file.) Note however that you cannot easily go from a canned format file back to a text file version—save the text file versions of your formats! Compared to input text files, formats have the additional advantage of speed, as we've mentioned, and of being articulated with the `Macros` menu.

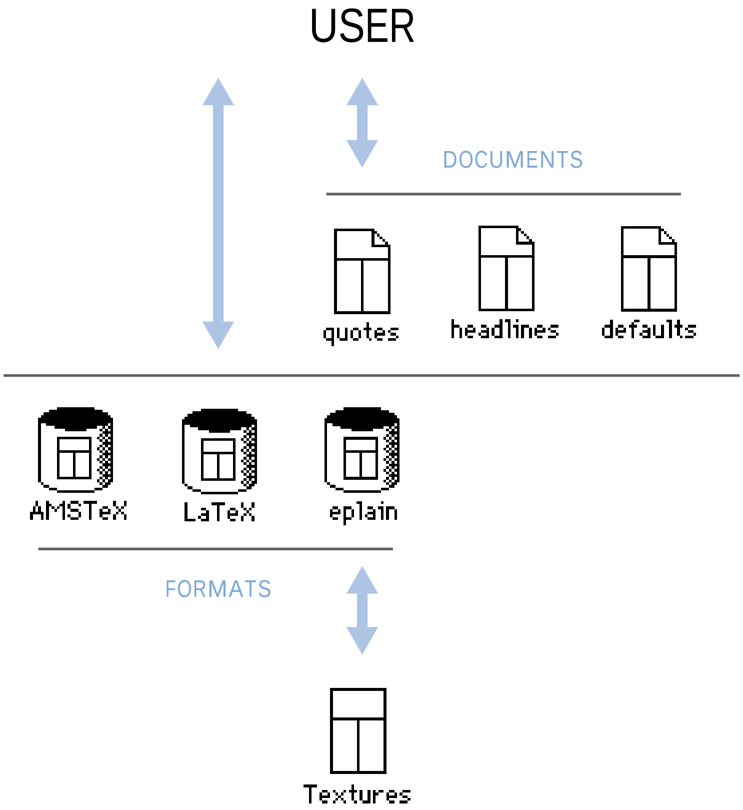


Figure 5-2. Formats built “on top of” TEX.

Plain TEX is the standard TEX format created by Donald Knuth and documented in *The TEXbook*. It is built into Textures. Plain TEX is a general format that can be extended for many uses, but it is only one of limitless formats, and any number of definitions may be added “on top of” Plain TEX to create a format for a specific application.*

In fact, you can build formats on top of formats, on top of formats... As illustrated in Figure 5-2, Plain TEX itself is built on top of TEX’s “**primitives**”—low-level control sequences that are not decomposable into simpler functions. There are about 300 of these “primitives.” Plain TEX adds another 600 commands to this basic set.

You can easily add new macros to Plain, or even change any of the Plain TEX control sequences to adapt it for specific needs (for

* You could even *replace* Plain TEX (by building upon the “VirTeX” format), though this will immediately get you into some “TEXpert”-level difficulties.

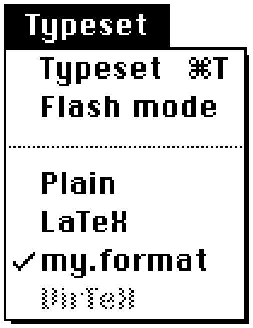
this, you should study *The TeXbook* to gain a fuller understanding of the workings of the Plain TeX format). A complete list of the Plain TeX control sequences appears in *The TeXbook*, Appendix B. Appendix E of *The TeXbook* contains examples of formats that can be added to Plain TeX for special applications.

IMPORTANT! *Formats such as LaTeX redefine some of the Plain TeX commands, so you can't simply typeset documents interchangeably under Plain and LaTeX.*

Selecting a Format

Whenever you typeset a file, one format will be used. ("No format"—TeX's primitive core only—is also specified as a format, called **VirTeX**, or "virgin TeX.")

The default format is Plain. To change to a different format, select the format in the Typeset menu. A check mark will appear beside its name, and the next time you typeset, that format will be active.



The format last used to typeset a given file is saved as part of that file's context information. To change the default setting (the initial setting for all new files you create), create a new file, select the format you want, and save your file giving it the name "Defaults" (as explained on page 70).

Adding a Format

To install available predesigned format files as well as format files you design yourself, simply drop the format file in the **TeX Formats** folder. To remove a format, remove the format file from the TeX Formats folder.

Changes to the TeX Formats folder take effect the next time you launch Textures.

Alternatively, from within Textures, you can *temporarily* install a new format with the File menu's Add FormatÉ item. Add FormatÉ displays a list of any format files on your disk; select a format file to add that format to the Typeset menu.

The LaTeX World

2.0 UPDATE: LaTeX mode in the Options Menu automatically initiates two complete typesetting runs to build .aux and crossreferences correctly. See page 17.

LaTeX ("LAH-tech" or "LAY-tech," as you will) is the major variant of TeX, and by itself accounts for a great part of TeX use around the world. LaTeX adds many new commands to the Plain TeX format, and also disables or modifies some existing Plain TeX commands. More highly structured than Plain, LaTeX defines a very general set of document styles. LaTeX consists of a set of macros and **style files** (which function as "sub-formats" for particular document types), together with a set of additional fonts. LaTeX is designed for producing large, structured documents such as journal articles, books, dissertations, or technical manuals, and includes features such as two-column layouts, automatic numbering of chapters, sections, and theorems, and much more.

LaTeX is documented in *LaTeX, A Document Preparation System, Second edition*, written by its creator, Leslie Lamport, and available from Blue Sky Research. The new LaTeX standard, LaTeX 2_ϵ, was released in 1994. To acknowledge the new standard and avoid confusion, we have adopted the following naming convention. LaTeX 2_ϵ is simply called LaTeX, and the older version of LaTeX is now called LaTeX 2.09. The Textures installer automatically installs LaTeX 2_ϵ, and provides LaTeX 2.09 as a Custom Installation option.

Plain vs. LaTeX In comparing Plain and LaTeX, the respective trade-offs are really two sides of the same coin, as summarized below.

Plain:

⊞ A nearly clean slate, unlimited flexibility.

⊟ A steeper learning curve: you need to do more to get started (you also need to write your own macros).

LaTeX:

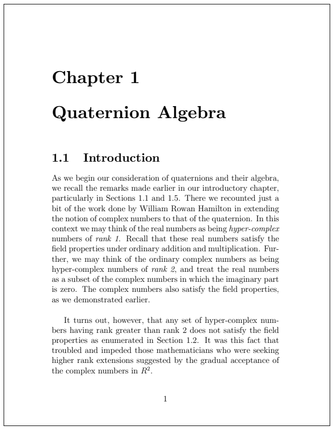
⊞ A good road for standardized material such as journal articles or books: an integrated set of macros is already provided for chapters, sections, indexing, tables of contents, and so on.

⊟ To *change* any of these in LaTeX can be awkward (or impossible).

Thus the essential trade-off to LaTeX is between not having to do it all yourself vs. the loss of flexibility that accompanies this.*

LaTeX Samples The LaTeX Samples folder (in the Samples folder) offers several views of what LaTeX can do:

- "Small2e" gives a very simple introduction to LaTeX.
- "Sample2e" is a general overview.
- "The Frog King" explains and demonstrates the LaTeX book format (as well as the BibTeX and MakeIndex tools, introduced below). To fully benefit from these samples, select `LaTeX` as your format (from the `Typeset` menu) and then typeset the sample files.



Chapter 1

Quaternion Algebra

1.1 Introduction

As we begin our consideration of quaternions and their algebra, we recall the remarks made earlier in our introductory chapter, particularly in Sections 1.1 and 1.5. There we recounted just a bit of the work done by William Rowan Hamilton in extending the notion of complex numbers to that of the quaternion. In this context we may think of the real numbers as being *hyper-complex* numbers of *rank 1*. Recall that these real numbers satisfy the field properties under ordinary addition and multiplication. Further, we may think of the ordinary complex numbers as being hyper-complex numbers of *rank 2*, and treat the real numbers as a subset of the complex numbers in which the imaginary part is zero. The complex numbers also satisfy the field properties, as we demonstrated earlier.

It turns out, however, that any set of hyper-complex numbers having rank greater than rank 2 does not satisfy the field properties as enumerated in Section 1.2. It was this fact that troubled and impeded those mathematicians who were seeking higher rank extensions suggested by the gradual acceptance of the complex numbers in R^2.

1

Figure 5-3. A journal article in the LaTeX report style.

* Another disadvantage of LaTeX, to speak Plainly, is that it doesn't necessarily let you take advantage of the beautiful and harmonious page layouts that you can achieve in TeX. (LaTeX books can tend to have a characteristic "utilitarian" look.) There *are* LaTeX style files that redesign the defaults for very nice results.

LaTeX-Related Tools: BibTeX and MakeIndex

BibTeX, and MakeIndex are public-domain tools designed to work with LaTeX.†

We've included on-disk, public-domain documentation where available; *LaTeX, A Document Preparation System* contains additional documentation. To demonstrate BibTeX and MakeIndex in use, we've also created a sample file called The Frog King (in the LaTeX Samples folder).

Bibtex helps with production of bibliographies. For documentation, see the BibTeX Notes folder and *LaTeX, A Document Preparation System*, Appendix B and Section 4.3.2.

MakeIndex creates indexes by using information generated by LaTeX's `\index` command. The MakeIndex Notes folder contains documentation.

Creating Your Own Format Files

You create canned formats with TeX's `\dump` macro. The `\dump` statement takes the place of a `\bye` or `\end` statement at the end of the file; it saves all of the macros that you have defined in your document as a separate format file. Creating a format is quite straightforward:

1. Make sure Flash mode is off. You cannot create a format file with Flash mode enabled.

2. Write your format file (or adapt the "preamble" of an existing text file), including your `\font` statements, `\def` statements, or whatever else you may wish to define. End the file with a `\dump` statement instead of a `\bye` or `\end`.

3. From the Typeset menu, select the format *to which you want to add* your format and then select the Typeset command. Your format is added "on top of" the format you use for typesetting: use Plain for an extension to Plain TeX, LaTeX for extending LaTeX with your own macros or styles, or VirTeX

† Public-domain macros are also available to let BibTeX and MakeIndex work with Plain TeX; the TeX-Index has more details. (See page 222.)

(“virgin TeX”) if you want to *replace* Plain TeX. You can also build formats on top of your own formats.

NOTE: *The TeXbook* says that `\dump` is “allowed only in INITEX.” This is *not* a concern in Textures: INITEX is a variant of TeX that is used to install TeX and create format files on some systems, but it is built into Textures and is therefore always available to you.

4. A dialog will ask you to save and name your new format. Save it in the **TeX Formats** folder and give it any name (you should not include the filename suffix “.tex”). The name will appear in your `Typeset` menu exactly as you name it.

5. Your format will now be available for typesetting the next time you start Textures. To get Textures to recognize your format for *immediate* use, take the additional step of adding the new format with the `Add FormatÉ` command.* Select your new format in the `Typeset` menu to typeset in the style it has defined.

That’s all. When you create a new format, its `Macros` menu will initially be empty. You can copy and paste an existing `Macros` menu and immediately use it, or better, customize or rewrite your old `Macros` menu to tailor your macros exactly to your new format.

NOTE: Hyphenation patterns can be added only by VirTeX (or by other formats that don’t have any hyphenation; not by Plain). Once added, hyphenation patterns cannot be extended by later definitions. The version of Plain that is included with Textures employs the American English hyphenation patterns; standard patterns are available for most other languages that use the Latin alphabet, and are available at various server locations (see page 221). If you need to alter a format by adding different hyphenation patterns (i.e. for a different language), you need first to select the `VirTeX` format, then `\input` the source of the format (for Plain TeX, the file **plain**, for LaTeX, the file **lplain**), making sure to replace also the **hyphen.tex** file with your own new file of hyphenation patterns. Then `\dump` the new format.

* If you are overwriting an existing format (that is, dumping under the name of an already existing format), the new format is immediately available, without any further action.

Recompiling Formats Written for Older Versions of Textures

If you are using format files that were compiled under older versions of Textures, you may need to recompile them to work under later versions of Textures. If this is the case, you'll see an immediate failure on typesetting and an error in the TeX log:

```
Incompatible format file:  Please \dump it again.
```

To recompile a format file, you'll need the format's source text file(s). You can then recompile the format file with TeX's `\dump` command and replace the old format file with the new one. (Recall that you cannot create a format file with `Flash mode` enabled.)

Specific instructions for recompiling LaTeX files are given on page 195.

PowerPC Formats

2.0 UPDATE: While you probably won't get the `Incompatible format file` message using 68K formats with Textures 2.0, it still won't work properly because many features of 2.0 do not function on 68K processors. See 19.

Format files are, alas, *not* compatible between the 68K and PowerPC TeX engines of Textures. If you have compiled formats from previous versions of Textures, it will be necessary to recompile them to get the performance advantage of the PowerPC processor. Older (68K) formats *will* continue to work, but *will not* run in PowerPC native mode. Because a PowerPC format can only be built by starting with a PowerPC base format, be sure to use the appropriate PowerPC base format (usually VirTeX or Plain) when you are creating formats. If you actually wish to create a 68K-compatible format, as when you're making a new "fat" format, you'll need to start from a 68K base format file.

Fat Formats

When you recompile an existing format, you can dump the new PowerPC format to the same container file as the older 68K format. If you dump to the same file, and answer "Yes" to replacing the existing file, this creates a "fat" format file that can then be used by both the 68K and PowerPC TeX engines. (This is an advantage only if you use both processors, otherwise it's just a waste of disk space.)

How To Identify PowerPC Formats Since the PowerPC Textures application can run both 68K and PowerPC formats, how can you tell which is running? (Other than the dramatic difference in speed, of course.) Answer: in the TEX log, the first line identifies the format name; if it's an older 68K format, it will say so here, for example "`LaTeX (68K)`."

We'll now take a first look at the baroque-going-on-rococo world of fonts on the Macintosh.

chapter 6

Introduction to Fonts

TEX's approach to fonts is completely independent of any particular type of computer or output device and doesn't encompass the descriptions of the actual characters themselves: this is left to the software and hardware of each TEX implementation. The Macintosh, for historical reasons, actually supports three different kinds of fonts, yielding a far from unified font system. This chapter gives an overview of these various font systems and their interaction. It serves as background for the following chapter, which gives the specifics of how Textures integrates the Macintosh fonts with the TEX system.

Because of the complexities of the Macintosh font system, you will probably need to understand something about the theory and practice of Macintosh fonts. If you already know about fonts in TEX and on the Macintosh, but only need to install fonts in Textures, you can skip directly to Chapter 7.

CONTENTS

Overview: Fonts of Type

In the tradition deriving from cast-metal typography, a **font** is a complete set of type all of a single size and style. With the advent of digital typography, the physical form of a font became less substantial—a representation in a computer file—, and the word *font* has come to be used in a looser and wider sense, sometimes encompassing multiple sizes.* Further, as we'll see below, what constitutes an individual font is defined somewhat differently in TEX and on the Macintosh.

Fonts are grouped into **font families** united by broad stylistic similarities. For instance, the Times font family includes the fonts Times Roman, Times Italic, Times Bold, and Times Bold Italic. The Computer Modern family, designed by Donald Knuth, includes 75 individual fonts.

For both TEX and the Macintosh, a single font is limited to a set of 256 characters.† (Standard font tables for TEX and the Macintosh can be found on page 208.)

Fonts on the Macintosh

On the Macintosh, fonts fall into three categories: bitmap fonts, PostScript fonts, and TrueType fonts.

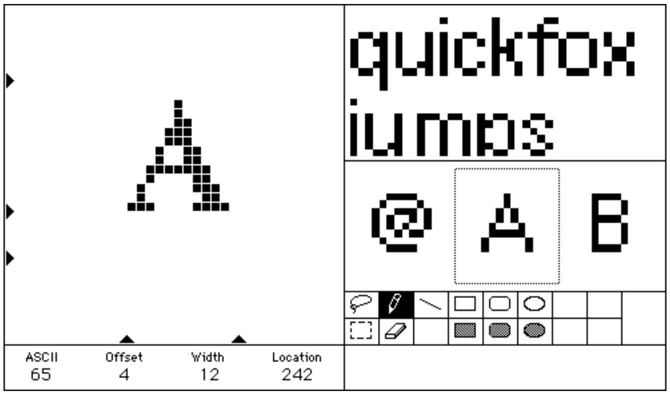
Part of the originality of the Macintosh, as opposed to older "command-line style" user interfaces, was its graphic interface wherein text was just a special case of graphics. The original Macintosh fonts, such as Chicago, Geneva, Monaco and New York, were **bitmap fonts**, stored as bit images in memory. A bitmap font consists of the bit images of every character in the font, and on the original monochrome Macintoshes, one bit in memory simply corresponded to one dot, or **pixel** on the screen (Figure 6-1).

Such bitmap fonts are easy to edit and fast to display, but they scale poorly, print slowly, and require lots of disk space.

* The word *font* is related to *found*, in its metallurgical sense (as in *foundry*): the word itself carries a sense of metal type, which is necessarily of a fixed size and style. Thus a font is the concrete representation—"the word made steel"—, as opposed to a *typeface*, which refers to the *design* of the type.

† Actually, the Macintosh also uses double-byte characters (yielding fonts of up to 28,000 characters) to represent characters in Chinese and Japanese, which have extremely large character sets.

Macintosh bitmap fonts are packed into font “suitcase” files (A) for distribution.



ASCII 65
Offset 4
Width 12
Location 242

Figure 6-1. Partial image of a bitmap font.

When the Apple LaserWriter was introduced, **PostScript printer fonts** were added to the Macintosh’s font set. Unlike bitmap fonts, PostScript fonts are **outline fonts**, described mathematically in Adobe Systems’ **PostScript** page description language, and therefore capable of arbitrarily high resolution (Figure 6-2). The LaserWriter’s read-only memory had the Times, Helvetica, Courier, and Symbol PostScript fonts built in; additionally, a person purchasing a LaserWriter received bitmap versions of these fonts for Macintosh screen display. Later, Adobe developed a system software add-on called the **Adobe Type Manager** (**ATM**), which allows PostScript fonts to display smoothly at any size on the Macintosh screen.

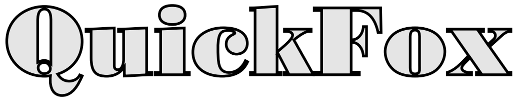


Figure 6-2. Characters in an outline font.

More recently, a collaborative effort between Apple and Microsoft has produced a third kind of fonts, **TrueType fonts**, introduced as part of System 7. TrueType fonts are Apple’s own alternative to Adobe’s PostScript fonts—like PostScript fonts, they are mathematically-defined outline fonts, but based on Apple’s QuickDraw graphics language rather than on Adobe’s PostScript language.

We’ll look at each of these font types in greater detail later in this chapter.

Fonts and Styles

In traditional typography, a typeface consists of a whole family of fonts: these might include Light, Italic Light, Roman, Demi-Bold, Heavy, Oblique, and so on. The Macintosh's original approach to fonts was cruder: a single, bitmap font was defined for each typeface, and "**styles**" —Bold, Italic, Outline, Shadow, Underline, Condensed— were applied to that base font. These styles were not individually designed and did not exist as individual fonts but were merely transforms of the base bitmap font, and therefore the results were not of high quality.

By comparison, outline fonts have come closer to traditional typography in that Bold, Italic, Bold-Italic, and sometimes other faces such as Demi-Bold are individually designed and defined as separate fonts, and are therefore capable of much higher quality. From the standpoint of the Macintosh user, bold, italic, and other faces continue to be selected from a font menu as "styles" of a "base" Roman font, but the underlying software structure is that of a family of fonts rather than of a single font. (Transforms are still applied if an intrinsic font is not found, or for styles such as underline that are not defined as separate fonts. These "derived" fonts are of acceptable but not superior quality.)

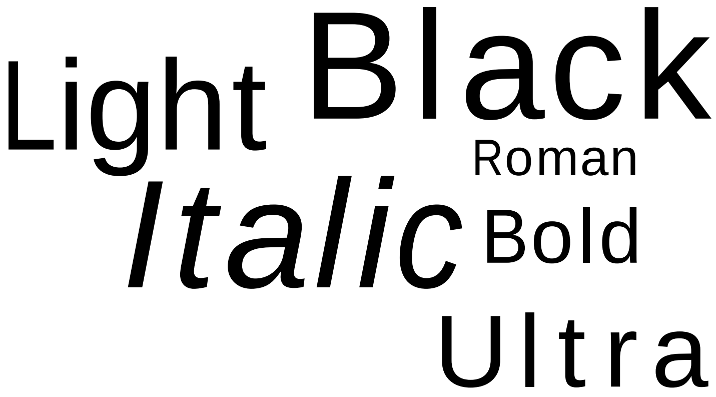


Figure 6-3. A family of typefaces.

Font Metrics

In addition to the character descriptions themselves, all fonts also include some kind of **metric information**. (This is invisible on the Macintosh; more visible and readily manipulable in TeX.) This metric information describes the size of each character. In the case of high-quality fonts, the metric information also describes how close various pairs of characters should be when they fall next to each other (this is called **kerning**), as well as **ligatures**: which characters should form compound characters when they fall next to each other in a word. (For example, *f* and *i* become *fi*.) TEX's high-quality typography demands very precise metrics, beyond the scope of what is needed for screen display or for draft-quality word-processing programs.

Fonts in TEX

For TEX, a font consists only of a font name, metric information, and mapping information. TEX's model of what is in a font (the **font map**) is an extension of the standard ASCII table, with each character in the font identified by an 8-bit character code, yielding a set of up to 256 characters in a single font.* Some font tables for TEX are shown on page 208.

To set type, TEX needs font metric information for each font that is used. In the TEX world, this information is called a **TEX Font Metric** (**TFM**). A TEX Font Metric contains three dimensions for each character: the character's height (above the baseline), its depth (below the baseline), and its advance width, as shown in Figure 6-4.

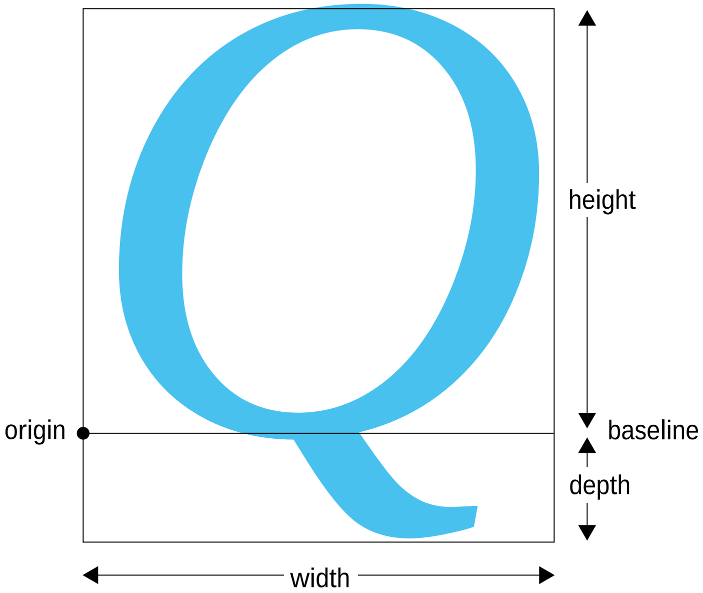


Figure 6-4. A character box: character dimensions in TEX.

* Actually, only 254 characters are really available on the Macintosh: ASCII 32 must be the space character and ASCII 13 is the carriage return character.

The metrics file may also include considerable information on ligatures and kerning, but not the character images themselves. In Textures, font metric files are identified by a special suitcase icon (🧳). To see a list of the fonts for which you have metrics installed, you can use the `Show Fonts` command from the `File` menu.

In fact, TeX itself needs *only* the metric information about a font: it "pastes" together boxes for each character, without having to know anything about what might be inside the box (Figure 6-5).

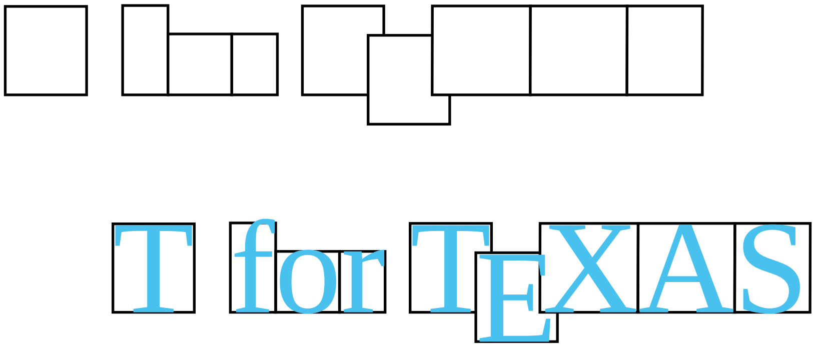


Figure 6-5. Character boxes.

From TeX's standpoint, it is up to each computer's operating system to provide the actual fonts—the actual character descriptions of what goes inside the boxes—so that characters may be viewed on the screen or printed. (In the case of the Macintosh, these may be either bitmap, PostScript, or TrueType fonts.) This is another example of how TeX itself is completely independent of specific computer operating systems or output devices. But though TeX itself is completely font-independent, by convention it *is* associated with the "Computer Modern" font family designed by Donald Knuth.

The Computer Modern Fonts

Computer Modern is the default font family in Plain, the format built into Textures. Created by Donald Knuth using his METAFONT type design system,* the Computer Modern fonts provide a very complete set of text and symbol fonts. (They also include full sets

* METAFONT is a programming language created by Donald Knuth that is capable of generating an entire family of fonts from a mathematical description. Nowadays, it's primarily used to modify the Computer Modern fonts themselves (for example, to reshape characters).

of ligatures and special characters that would be called “expert fonts” and sold separately in the case of other high-quality font families.) Computer Modern fonts are designed for general text or body copy; the family also includes several fonts of special characters and symbols for mathematical, scientific, and technical use. Textures includes the full set of 75 Computer Modern typefaces, including the 16 faces in the Plain TEX set, although some of these are rarely used.†

The Computer Modern fonts in Textures are in Adobe PostScript Type 1 form. These fonts can be used with all PostScript printers, as well as with QuickDraw printers such as a Hewlett-Packard DeskWriter (provided you install the Adobe Type Manager). These fonts can be used with all Macintosh applications.

Computer Modern is also known as CM, and all of its font names start with “**cm**”. The letters that follow the “cm” indicate the sub-family, and subsequent digits give the design size in points. For example, “cmr10” means Computer Modern Roman at 10 points. The *Computer Modern Faces* booklet that you received with your Textures package shows the characters of each font. For an exhaustive description of the Computer Modern faces, see Volume E of *Computers and Typesetting* by Donald Knuth (Addison-Wesley, 1986). See also Appendix F of *The TEXbook*.

Like other PostScript fonts, Computer Modern fonts come in three pieces: the font metric resource (built into the Textures application), a PostScript printer font, and a bitmap screen font. The CM fonts are automatically installed by the Textures Installer program.

NOTE: Only one bitmap screen font is installed for each of the Computer Modern fonts. On the screen, the font will display properly at its design size (which is the same as the screen font size); but if the font is scaled, it will display poorly on the screen *unless* you install the Adobe Type Manager (ATM). Screen fonts and ATM are explained below on pages 128 and 131.

† The following CM fonts are particularly rarely used: **cmdunh**, **cmff**, **cmfi**, **cmfib**, **cminch**, **cmu**, **cmvtt**. Removing them from your system is unlikely to cause you any problems. (Of course, you should save backup copies of all of your fonts, just in case you need to print a document that uses one of them.)

NOTE: Because there are so many CM fonts, you may find that they crowd your applications' menus. There are several ways to reduce the number of fonts in your menus, or to make them available only when you are in Textures. See page 142 for details.

Plain TeX Default Fonts

The Plain TeX format has macros that automatically reference sixteen of the Computer Modern fonts. For example, in the Plain TeX format, the command `\rm` will produce typeset output in Computer Modern Roman at 10 points (this is the default if nothing is specified). Typing `\it` switches to Computer Modern Text Italic at 10 points. You can easily redefine these Plain TeX commands to specify any other fonts. See *The TeXbook*, Chapter 4, for more on the commands that produce output in these Computer Modern faces.

To use any other fonts—whether in the Computer Modern family or not—you must first define the font with Plain's `\font` statement; you can then invoke it for use later in the file. For example, the command

```
\font\tenit=TimesI at 10pt
```

redefines "`\tenit`" to specify Times Italic rather than Computer Modern Text Italic. You can then switch to this font at any point in the file by entering the command `\it`. (Use braces to constrain the effect of a font change; for example, `{\it texto italicizado}` yields *texto italicizado*.) It's good practice to place all of your `\font` commands together at the beginning of a TeX file.

You can also build `\font` commands into a "canned" or precompiled format of your own using TeX's `\dump` macro. (See page 111.)

We'll now look at bitmap and outline fonts on the Macintosh in greater detail.

Bitmap and Outline Fonts

Perhaps the easiest way to understand a bitmap font is to imagine a grid on which we've drawn a letter, as shown in Figure 6-6.

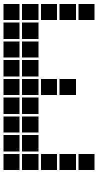


Figure 6-6. Rasterized image of a character.

To have this character appear as a recognizable “E” on either the screen or the printer, we must decide which squares, or pixels, in the grid must be black. This process is referred to as **rasterization** or **rendering**.

Bitmap fonts store a description of each character as a bit image in memory. Every font must be bitmapped at some point, either on screen or on the printer. Both the screen and printer will unavoidably possess a grid—72 dots per inch (dpi) on the screen, 300 to 600 dpi on a laser printer, and 1200 to 3000 dpi on a phototypesetter—, and a font of whatever sort must accommodate itself to that grid. What distinguishes a bitmap font from an outline font is that the characters are not only rendered but also *stored* as bit images.

Because bitmap characters are stored as bit images (i.e., the rasterization has already taken place), the computer need only recall the bit image from memory in order to display characters from the font—a relatively fast process. But bitmap fonts have two big drawbacks. First, bitmap fonts take up a large amount of disk space. This is because a bitmap character image is specific to both the point size and the resolution of the output device: you need a separate bit image at each size and output-device resolution you might want to use. A 10-point screen font, for instance, at the screen resolution of 72 dpi would be a separate bit image from a 10-point bitmap font for a printer resolution of 300 dpi. Second, if you want to use the font at a resolution or at a size that you don’t have in memory, the font will have to be scaled and the result will usually be bad (Figure 6-7).

Due to their inefficient storage and poor scaling, bitmap fonts have been relegated to a screen-display role on the Macintosh and increasingly, replaced altogether by outline fonts. (In the case of PostScript fonts, at least one bitmap font must be present so that the Macintosh operating system can “find” the corresponding outline font. No bitmap fonts are required for TrueType fonts.) For printing, PostScript and TrueType outline fonts are preferred.*

An **outline font** contains only a geometric outline description of each character, represented as a series of lines, curves and points. This requires relatively little disk space, since no attempt has yet been made to rasterize it for a particular output device (that is, lay it on a specific grid and decide which pixels should be black). In effect, an outline font "knows" the *shape* of the character, rather than just knowing which pixels to turn on for a particular character at a particular size. When it renders the character on screen or on the printed page, it applies this ideal shape to the appropriate grid and creates as smooth a rendition as the medium will allow. Outline fonts can be scaled, rotated, and transferred from one output device to another with little or no loss of quality.

When an outline font is rasterized (bit-mapped) for output, **hinting** may or may not be applied. Hinting means extra information to help in fitting the "noumenal" world of abstract (perfect) curves to the phenomenal world of actual (imperfect) bits. For example, consider a case where the ideal requires 2.5 pixels, so that either two or three pixels might be used to fill out the width of a line. Hinting information might specify that this line should be the same as some other line in the character, so that the rasterizer knows whether to round up or round down if it's a toss-up between 2 pixels or three.

Design Sizes and Scaling

Every bitmap font has an explicit *design size*, and other sizes will be scaled directly from the design size. For example, a 12-point bitmap font places pixels to get the best possible representation of a character at 12 points (and at the given output resolution; e.g., 72 dpi). If you scale that 12-point font to 20 points, some sort of scaling algorithm will be applied, and it can only approximate a good representation at 20 points. When a bitmap font is scaled, its visual quality will deteriorate dramatically, yielding a jagged and "stair-stepped" image (Figure 6-7).

* At low (i.e., screen) resolution, hand-tuned bitmap fonts actually give the best results, but it is time-consuming to create such fonts.

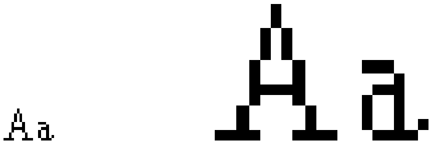


Figure 6-7. Scaled bitmap font.

Outline fonts, on the other hand, are drawn well at all sizes, especially large sizes. But outline fonts too have an implicit design size —that is, a point size where the font design fits best. When an outline font is scaled linearly, every element of the character changes in direct proportion to the point size so that even though the image is crisp, the proportions may become awkward. The "best fit" for most of the standard Adobe fonts, such as Times, is roughly 18 points, a compromise size so that the font scales relatively well for both body type and headline sizes. But with linear scaling, when you scale your "18-point" Times Roman to 6 points, the same *shape* will be drawn at 6 points, just smaller. This will not always work on a practical aesthetic level, since a font designed for use at one size may look unnatural at others, as illustrated in Figure 6-8. (Adobe's "TimesTen" font responds to this problem: it is a Times font with a design size of 10 points, more suitable for body-type sizes.)

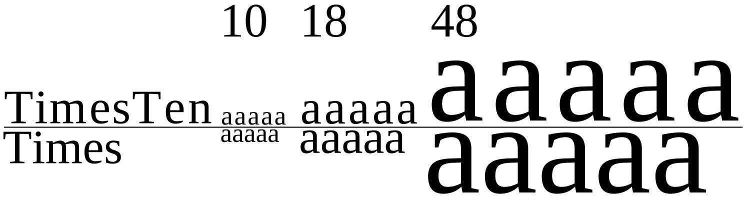


Figure 6-8. Design sizes: Times and TimesTen fonts scaled to large and small sizes.

Donald Knuth hand-designed the Computer Modern fonts for a variety of sizes; thus for a Roman face, we have cmr5, cmr6, cmr7, and so on, yielding a quality beyond what you can get by linear scaling.

PostScript Fonts

PostScript fonts, written according to the font specifications published by Adobe Systems, are built into, or downloadable to, PostScript printers such as the Apple LaserWriter. A set of 13 PostScript fonts comprising the Times, Helvetica, Courier, and Symbol font families was built into the read-only memory of the original LaserWriter. To this set, LaserWriter Plus models added another 22 fonts. These 35 PostScript printer fonts became the de facto printer font standards in the Macintosh world, and font metrics for all of them are built into Textures.* The standard PostScript printer fonts you encounter are **PostScript Type 1** fonts.†

To print with PostScript fonts, you need either a PostScript printer or the Adobe Type Manager (ATM) software and a QuickDraw (non-PostScript) printer. As Figure 6-9 illustrates, when you use a PostScript printer, Textures sends the message to image a character directly to the PostScript interpreter in the printer itself. With a non-PostScript printer the rendering process is done by your Macintosh. In either case, the character outline is adjusted to the proper size (e.g., 18-point) and then rasterized according to the resolution of the current output device (e.g., 300 dpi).

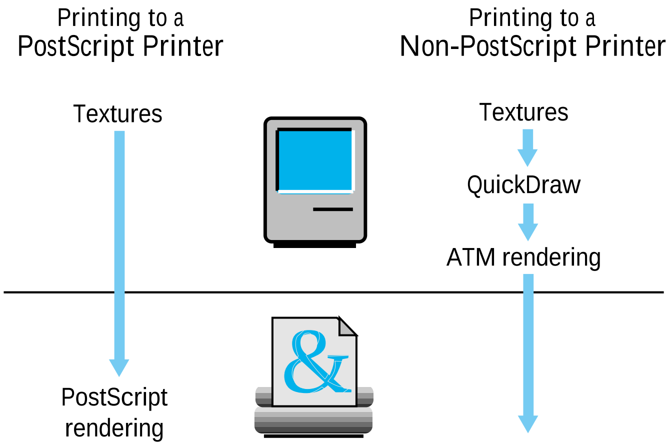


Figure 6-9. Macintosh–LaserWriter relation.

* Because TEX needs only the font's metric information, not its character images, it can "set type" for any fonts for which it has metric information (that is, fonts that are listed in Textures' Show Fonts dialog)—even though the actual fonts themselves may not be available on your system. However, to view the proper fonts on the screen or to print them, you must have the actual fonts installed in your system.

† "Type 1" was originally a private Adobe format, and the number of Type 1 fonts was therefore limited. Adobe offered a public format, Type 3, but these fonts were not hinted. The Type 1 format has since been made public, and almost everyone who designs fonts now uses this format. Type 2 fonts are "multiple master" fonts: fonts with more than one design axis.

Downloadable PostScript fonts (fonts that don't permanently reside in the printer's ROM) are stored in printer RAM memory, and it can require substantial printer memory to store these fonts. (This is also true of TrueType fonts.) Only a portion of the printer's memory is allocated to storing fonts, and even if you plan to use only one character from a font, the *entire* font is downloaded to the printer.

A PostScript font actually comes in two pieces:

a bitmap **screen display font**, contained in a font "suitcase" file. This is a Macintosh bitmap font as described in the previous section. Each PostScript font comes with at least one bitmap font; it is in a sense auxiliary to the PostScript printer font and is designed for screen display only.

a PostScript **printer font**. This is the outline font that the PostScript printer uses to create the image you ultimately see on the paper. This is what people are normally referring to when they speak of PostScript fonts.

This outline font is also what the Adobe Type Manager (ATM) uses to render output for your screen display or for a non-PostScript printer in case you don't have a bitmap font available at just the right size and resolution for that purpose.

Additionally, metric information about each font is contained in an **Adobe Font Metrics** (**AFM**) file. Although other Macintosh applications don't take advantage of this precise metric information, it is indispensable to Textures, as we'll see in the next chapter.

An awkward feature of the Macintosh font system is that the screen imaging model, QuickDraw, is separate from the PostScript imaging model that has become the standard for high-quality printing and phototypesetting. A consequence of this dualism is that PostScript's quality font-scaling capabilities are not directly available to the Macintosh system software for displaying fonts on the Macintosh screen, and scaled fonts therefore look bad. The Adobe Type Manager is a system add-on that corrects this problem.

PostScript Screen Fonts with the Adobe Type Manager (ATM)

ATM is a PostScript "font processor" that resides in Macintosh RAM. When you are using a PostScript font but don't have a corresponding bitmap font of the appropriate size and resolution for the display, the Adobe Type Manager reads PostScript outline fonts ("printer fonts") and uses that information to render characters for your screen display. ATM can also render characters for any other non-PostScript output device, including QuickDraw printers such as the Hewlett-Packard DeskWriter. (With ATM, the print quality on a non-PostScript printer is excellent.) In regard to text (but not graphics), ATM thus does for your non-PostScript device the same thing the PostScript interpreter does inside a PostScript printer, but operating in Macintosh memory rather than in printer memory. ATM works automatically in the background without your intervention. It has no effect on the display of bitmap or TrueType fonts.

NOTE: Current versions of ATM can be obtained from Blue Sky Research or directly from Adobe Systems, Inc. for a nominal charge; see our web page, www.bluesky.com or page 224 for details.

You can use Textures successfully without the Adobe Type Manager: characters in your Typeset window will display poorly at sizes other than the size of the screen bitmap font, but output on a PostScript printer will not be affected. You *do* need ATM to properly print the Computer Modern, AMS, or any other PostScript fonts on a non-PostScript printer. If you use the Computer Modern fonts without ATM, choosing a viewing magnification that matches your available screen fonts improves both speed and ease of viewing.

The ATM Font Cache Since ATM uses Macintosh memory (RAM) to render fonts for display or printing, excessive font rendering can slow your system's performance. However, once ATM renders a font, that information is stored in a dedicated memory area called the ATM font cache: if characters from the font are needed again, they are simply recalled from the font cache. If you find that your computer is performing too slowly after installing ATM, try increasing the font cache. If you use the Computer Modern or AMS PostScript fonts with Plain TEX, you should set the font cache to at least 256K. The LATEX or the AMS preprint styles use significantly more fonts than Plain TEX, and may work better with a font cache of 512K or more.

TrueType Fonts

TrueType fonts are a PostScript “knock-off”—Apple’s answer to PostScript—, and they bring Macintosh fonts up to PostScript quality. Like PostScript printer fonts, TrueType fonts describe the outline of a character, adjust it to the proper size, and then rasterize it according to the resolution of the output device. But where PostScript fonts are processed by a PostScript interpreter, normally in the printer, TrueType fonts are sized and rasterized by the Macintosh system software before the image is sent to the printer. (This is analogous to PostScript fonts that are sized and rasterized in Macintosh RAM by ATM in the case of a non-PostScript printer.)

TrueType fonts offer an advantage in ease of handling, in that their processing is built into the system software and ATM is not required. Otherwise, TrueType fonts have no particular advance over the PostScript technology. (For Apple, they have the advantage of not belonging to an outside vendor.)

You can presently use TrueType fonts, such as Times, for which corresponding PostScript metrics are available in Textures.

chapter

7

Fonts in Textures

We've already seen that Textures provides the bridge between the standard TEX system and the multi-fonted Macintosh world. In this chapter we'll look at the specifics of how fonts work with Textures. We'll begin by looking in more detail at the distinction between *fonts themselves*—the actual character descriptions—and *font metrics*, which describe how those characters fit together. To install additional fonts in Textures may require you to create new font metrics files, and we'll see how you can do this with a Textures utility called EdMetrics.

2.0 UPDATE: Font usage has been automated in Textures 2.0, allowing you to use installed fonts with Textures without creating metrics. Font style modifiers, like bold, and italic can be added interactively in the Edit Window. See page 17.

CONTENTS

Textures Fonts and Font Metrics

2.0 UPDATE: Textures now automatically creates font metrics for any fonts installed on your Macintosh. See page 17.

In the previous chapter, we saw that TeX typesets using a font's **metric information** only, leaving it up to each installation to provide the actual fonts and printing facilities. Textures provides the interface between TeX's "metrics-only" font system and the concrete world of Macintosh fonts via **font metrics resources**. (When we refer to *font metrics* below, we mean font metrics resources.) Font metrics resources are stored in one of two places:

Textures

In the Textures application itself, in the case of the commonly-used fonts listed below in Table 7-1.

metrics

In Textures **font metrics suitcase** files, kept in the TeX Fonts folder.

A font's metric information consists of:

- *Metrics information* per se: minimally, the height, depth, and advance width of each character in the form that TeX requires. There may also be considerable information on ligatures, kerning, and other metrics.

- *Character mapping information* so that TeX can use fonts from the Macintosh and PostScript font systems. In TeX's internal model of a font (and in the CM family), some ligatures and special characters are placed at positions different from what is standard in the Macintosh or PostScript font systems; so Macintosh or PostScript fonts must be remapped from the font model that TeX uses.*

* For instance, the CM fonts generally make full use of the first 32 positions in the ASCII table, but in the Macintosh system characters placed in these positions will cause problems, so these characters must be re-mapped to positions above 128. Font mapping is discussed further in Appendix B (page 204).

Metric information for more than 120 fonts is currently built into the Textures application itself. As shown in Table 7-1, these built-in metrics include the standard PostScript printer fonts, all of the Computer Modern fonts, and all of the LaTeX fonts, so you won't need separate metrics files for any of these fonts. *Any other fonts* will require their own metric information in order to work with Textures; this metric information will be contained in a separate font metrics "suitcase" in the TeX Fonts folder.

NOTE: Font metrics suitcases contain the Textures equivalent of TeX Font Metrics (TFM) files on other TeX systems. On traditional TeX systems, a TFM file has information on only one font, but in Textures, a font metrics suitcase may contain the metrics resources for multiple fonts.

To determine which metrics are already available on your system, use the `Show Fonts` menu item.

Table 7-1. Font metrics built into Textures itself.

Plain TEX CM metrics

cmbx10/7/5
cmex10
cmmi10/7/5
cmr10/7/5
cmsl10
cmsy10/7/5
cmti10
cmtt10

LATEX CM metrics

cmbsy10
cmbx9/8
cmcsc10
cmmi9/8/6
cmmib10
cmr9/8/6
cmsl9/8
cmss10/9/8
cmsy9/8/6
cmti9/8/7
cmtt9/8

Metafont logo metrics

logo10/9/8
logobf10
logosl10

Other CM metrics

cmb10
cmbx12/6
cmbxsl10
cmbxti10
cmdunh10
cmff10
cmfi10
cmfib8
cminch
cmitt10
cmmi12
cmr12/17
cmsl12
cmsltt10
cmss12/17
cmssbx10
cmssdc10
cmss10/12/17/9/8
cmssq8
cmssqi8
cmtcsc10
cmtex10/9/8
cmti12
cmtt12
cmu10
cmvtt10

LATEX symbol metrics

lasy10/9/8/7/6/5
lasyb10
lcircle10
lcirclew10
line10
linew10

PostScript metrics

AvantGarde/B/BI/I
Bookman/B/BI/I
Courier/B/BI/I
Chancery
Dingbats
Helevetica/B/BI/I/O
HelveticaN/B/BI/I
Palatino/B/BI/I
Schoolbook/B/BI/I
Symbol
Times/B/BI/I

CM metrics: all 75 fonts of the Computer Modern family (introduced on page 123). The Plain CM metrics are standard fonts referenced by the Plain format; LATEX CM metrics are *additional* CM fonts referenced by the standard LATEX format and styles, while the Other CM metrics are not referenced by any predefined formats. Individual documents are not likely to require all fonts referenced by their formats, but additional font metrics may be required by additional definitions in any document or format.

PostScript metrics: the standard Macintosh set of 35 LaserWriter fonts. For most font families, these include a “plain” (Roman) font, together with bold (“B”), italic (“I”), and bold-italic (“BI”) fonts. The sans-serif Helvetica family (useful for titles and labels) also includes narrow (“N”) and outline (“O”) fonts.

LaTeX Symbol metrics: this is the complete set of eleven LaTeX symbol fonts.

“logo” metrics. The “logo” fonts were expressly designed to print the word METAFONT, and consist only of the seven characters that appear in this name. (The Textures Installer doesn’t install this font.)

You can create and edit your own Textures font metrics with a utility program called **EdMetrics**, which is provided in the Font Tools folder.

Fonts Provided with Textures

As for the actual fonts themselves (that is, the files that contain the actual character descriptions), the full set of Computer Modern, American Mathematical Society and LaTeX fonts is shipped with every version of Classic Textures. (For a depiction of each CM font and AMS font, see the Textures Documentation folder on the Textures CD-ROM. The following fonts are also available from Blue Sky Research for separate purchase:

- **MathTime fonts**: these fonts contain a set of mathematical symbols designed to harmonize with the standard Times font family. They form a complete replacement for the CM math fonts for those who prefer Times. Because the Times font is built into the LaserWriter ROM, the MathTime fonts have the additional advantage over CM that they require far less printer memory.

- **Lucida Bright** and **Lucida New Math**: these are part of a family of harmonized typefaces by Bigelow & Holmes that can serve as a complete replacement for the CM fonts.

For information on ordering these fonts, see page 224 in Appendix D.

Show Fonts Dialog

The File menu’s Show Fonts command lists the fonts for which Textures has metric information. If you select the Only fonts in use checkbox, it will show you what fonts are used by the active document. (Clicking the dialog's OK button simply closes the dialog box—`Show Fonts` has no effect on your document or display.)

To see whether a font is actually installed on your system (that is, available for screen display or printing), select the name of the font in the `Show Fonts` dialog. If the font is installed, the boxes titled `Screen` and `LaserWriter` will show the sizes at which the font is available for screen display and printing on the LaserWriter. Another box displays a partial alphabet in the selected typeface and at the selected size (Figure 7-1).

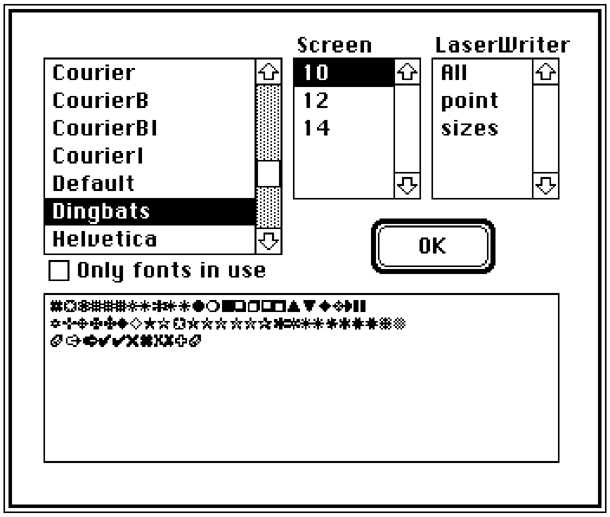


Figure 7-1. Show Fonts Dialog.

If the font is not installed, the dialog's message box will display the message:

```
[No face installed.]
```

If you typeset using a font that is not installed, the display in the Typeset window will substitute characters from the Chicago font, and printed output will be in Courier (on a PostScript printer) or in Chicago (on a QuickDraw printer). As we've mentioned, TeX itself requires only the font's metric information in order to typeset. This means that TeX can set type correctly for fonts that are not installed in your system, even though characters in the font will not display or print properly on your machine.

NOTE: If you look at the `Show Fonts` listing, you'll see a metric called "Default." This metric (actually Times Roman) is used by TeX for screen display purposes when the requested font's metrics are unavailable; the resulting type will appear grayed out in your Typeset window.

You can use the Add FontsÉ command to temporarily add more fonts or metrics to your current set; see page 149.

The TeX Fonts Folder

The TeX Fonts folder should be in the Textures folder (i.e., in the same folder as your Textures application), where Textures will automatically find it. The TeX Fonts folder is the place to store Textures font metrics suitcases, so that font metric information will be available to Textures during typesetting.

Every time you start Textures, Textures automatically installs all of the font and metrics suitcase files present in the TeX Fonts folder.

The Textures Installer program automatically places all of the CM and LaTeX fonts in the appropriate places in the System Folder. The only files that are initially placed in the TeX Fonts folder are the invisible fonts and Slitex metrics used with Slitex.

Installing Fonts in Textures

The Textures Installer installs all of the Computer Modern, AMS, LaTeX, and SliTeX fonts. As we've mentioned, Textures already has built-in metric information for the standard PostScript LaserWriter fonts, so they are immediately available for use (assuming you've installed the fonts according to the directions in your Macintosh documentation). If you are installing additional fonts, this section explains where they should go.

When you install a new PostScript font, there are actually *two* fonts to install:

The first piece of a PostScript font "team" is the bitmap **screen display font**. The Macintosh system software requires this piece not just for screen display, but in order to find the PostScript outline font. Bitmap screen fonts are shipped in font suitcase files. (These are *not* the same as Textures font metrics suitcases!)

The second piece is the PostScript **outline font** itself (also called a *printer font*).

When you get a PostScript font, you will get files both for the screen fonts (likely all stored in a single suitcase) and for each outline font (stored as individual files).

With System 7.1, and later, font installation has been greatly simplified. If you are running an earlier version of system software, contact Blue Sky Research technical support for assistance (see page 224).

There are two steps to installing a PostScript font for which Textures lacks font metric information:

1. Install the screen bitmap font and the PostScript outline font.
2. Install the font's metric information.

We'll now step through these processes.

Where to Install Bitmap Fonts and PostScript Fonts

With System 7.1 or later, you can simply drop the bitmap and postscript fonts on the closed System Folder and they will be installed in the Fonts folder.

Where to Install Textures Font Metrics

Textures font metric "suitcase" files must be installed in the TeX Fonts folder (or in the Textures application itself) to allow Textures to find them during typesetting. If you are installing a PostScript font whose metric information is not built into Textures, place the metric suitcase in the TeX Fonts folder.

If you don't have Textures font metrics for the font you are installing, you'll also need to convert the font's AFM (Adobe Font Metrics) file into a Textures font metric, using EdMetrics.

Making Fonts Available only to Textures

If you want the Computer Modern and AMS fonts to appear in your menus only when you are in Textures but not when you are using other applications, run the Textures Installer on the Textures CD-ROM, and select Custom

Installation of the `Hidden Computer Modern and AMS Fonts`. See the *Textures Installation Guide* for more information.

Creating New Font Metrics with EdMetrics

2.0 UPDATE: Font style modifiers, like bold, and italic can be added directly into the font command in the Edit Window. See page 18.

2.0 UPDATE: Existing font metrics can now be directly accessed through Show Fonts, by double-clicking on a selected font name in the Show Fonts dialog box 18.

EdMetrics is a tool for creating and editing Textures font metrics "suitcase" files. After you've installed a new font, you can install its metric information with EdMetrics. EdMetrics can read metric information from three kinds of metrics files:

- Adobe Font Metrics (AFM) files for PostScript fonts.
- Property List (PL and VPL) files from other TEX systems.
- Macintosh FOND resources for QuickDraw fonts.

It then converts that information into Textures font metrics.

Creating PostScript Font Metrics

Each PostScript font should come with an **Adobe Font Metrics** (**AFM**) file that contains the metric description of the font. Although other Macintosh applications have not taken advantage of the precise metric information made available in AFM files, Adobe Systems was looking ahead to high-precision typography when they designed the PostScript font specifications, and the metrics in AFM files are fully sufficient to the high demands of TEX.

When you install a new PostScript font whose metrics aren't built into Textures, you can create the Textures font metrics for them as follows:

1 Make sure you have the AFM file for the font. (It's on the font disk from the vendor.) If the AFM is not available, you should get it from the font vendor or from Adobe Systems.

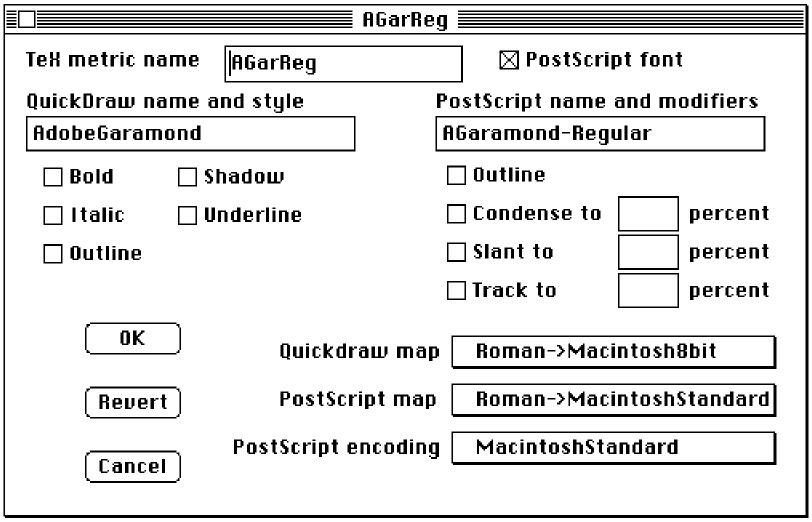
2 Launch EdMetrics. (EdMetrics is contained in Textures' Font Tools folder.)

3 Use EdMetrics' `New` command to create a new metrics suitcase, and give it a name (e.g., "Garamond Metrics"). You can also open an existing metrics suitcase and add your metric to that, if you prefer.

4 From EdMetrics' `File` menu, select `Add AFM MetricÉ` and then specify the AFM metric to be added.

NOTE: The `Use CM Layout` item in EdMetrics' `File` menu is checked by default and should be on for normal text faces. This means to use the standard TEX mapping for a text font (called "Roman" or "TeX Text"); this is the font layout given in Appendix F of *The TEXbook*, page 427. (See Appendix B, "Font Mapping," beginning on page 204.)

5 EdMetrics' main (metric-mapping) dialog will appear, as shown in Figure 7-2. (On the following pages, we'll look in detail at the various elements you can specify for a font metric; for now we will summarize the process.)



AGarReg

TeX metric name: AGarReg

☒ PostScript font

QuickDraw name and style: AdobeGaramond

☐ Bold ☐ Shadow

☐ Italic ☐ Underline

☐ Outline

PostScript name and modifiers: AGaramond-Regular

☐ Outline

☐ Condense to ___ percent

☐ Slant to ___ percent

☐ Track to ___ percent

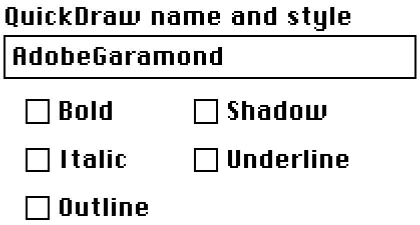
Quickdraw map: Roman->Macintosh8bit

PostScript map: Roman->MacintoshStandard

PostScript encoding: MacintoshStandard

Figure 7-2. EdMetrics main dialog.

A In EdMetrics' main dialog, give a `QuickDraw name` to the font if necessary. This is the name that will appear in font menus.



QuickDraw name and style

AdobeGaramond

☐ Bold ☐ Shadow

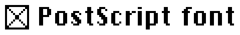
☐ Italic ☐ Underline

☐ Outline

NOTE: If the `QuickDraw name` field shows a garbled name, it likely means that you have not yet installed the font (as explained above starting on page 141). You should install the font before editing the font metrics.

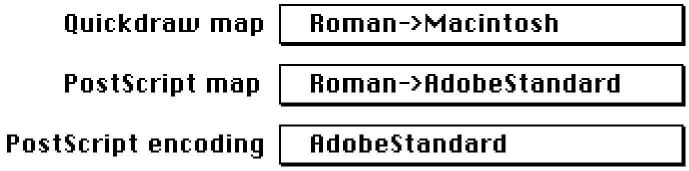
B In QuickDraw, bold and italic are selected as "styles" of a font.

C The `PostScriptfont` check box should be checked: this indicates that the font is an outline rather than a bitmap font. (This box should be checked for TrueType fonts also.)



☒ PostScript font

D Specify the **font mapping**: this is the mapping of characters from TEX's font map to QuickDraw and PostScript for output to screen or printer, telling Textures where to find ligatures and so forth.



Quickdraw map Roman->Macintosh

PostScript map Roman->AdobeStandard

PostScript encoding AdobeStandard

For normal text fonts, you should specify a PostScript mapping of `Roman->Adobe Standard`. The PostScript encoding should correspond to the PostScript map (that is, "AdobeStandard" for a PostScript map of Adobe Standard). For the QuickDraw mapping, you would normally specify `Roman->Macintosh`. More details on font mapping are given in Appendix B, beginning on page 204.

6 Click the OK button.

NOTE: Add AFM MetricÉ actually performs two distinct actions: first, it adds the AFM (i.e., creates a Textures font metric from the AFM); and second, it opens the new metric for editing. If you click the Cancel or Revert buttons, it is the second step—i.e., any editing changes—that is cancelled or reverted.

The new font metric is now added to your font metrics file. For each font that you are adding, repeat steps 4 and 5.

7 Place your font metrics file in the TeX Fonts folder.

IMPORTANT! DO NOT drop your new font metrics suitcase file into the System Folder—your metrics will disappear and you will need to reinstall (or ResEdit) your System File to get rid of the “garbage” information you have dumped into it.

Your font is now fully installed for use with Textures. To typeset with the new fonts, restart Textures or “open” them with the Add FontsÉ command.

Note on Font Names Unfortunately, the different font systems also have different rules for naming fonts:

- **Textures/TEX font names**: No spaces are allowed, and names are not case-sensitive.
- **Macintosh (QuickDraw) font names**: Names are not case-sensitive and may be up to 32 characters in length. (This is the name that will appear in your font menus.)
- **PostScript font names**: Names are *case-sensitive* and may be up to 64 characters in length. No spaces are allowed.

By convention, Textures gives each PostScript font a one-word family name, with suffixed letters for different styles: “B” for bold, “I” for italic, and “BI” for bold italic. For example, “Times-Roman” in PostScript is “Times” in Textures, and “Times-Italic” in PostScript is “TimesI” in Textures. Oblique is conventionally equivalent to italic, so “Helvetica-Oblique” becomes “HelveticaI.” In addition, the suffix “O” means outline, as in “HelveticaO”, while “C” means condensed, as in “GaramondC.” The various font names are entirely arbitrary and you can change any of them with EdMetrics. You can also duplicate a metric and change its name to be compatible with other font naming conventions.

Creating Macintosh (QuickDraw) Font Metrics

2.0 UPDATE: This process has been automated in Textures 2.0. See page 17.

Adding a QuickDraw font metric is like adding an AFM metric, as described in the previous section, except that in step 4 above you choose `Add FOND MetricÉ` rather than `Add AFM MetricÉ`. On the Macintosh, font metric information is contained in a **FOND** (font family) resource. FONDs are used for both bitmap and TrueType metrics.

The `PostScriptfont` box should also be checked for TrueType fonts.

The standard text-font mapping is `Roman->Macintosh` . (PostScript mapping is irrelevant in this case.)

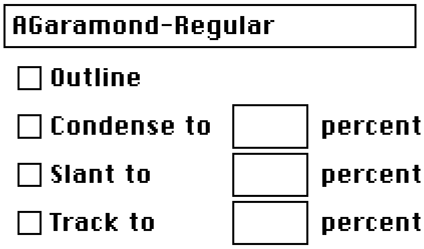
IMPORTANT! Much of the metric information in a FOND resource is optional under Apple's specifications, and a FOND does not necessarily contain sufficient information for successful installation. The metric information in AFM metrics (for PostScript fonts) or PL metrics (from other TEX systems) is more complete than the metrics found in Macintosh FOND resources, and it is preferable to make a Textures metric from an AFM or PL, if available. In particular, FOND resources may not include ligature and kerning information, and the "bounding box" information in a FOND is not as precise or as consistently used as the dimension information that TEX optimally requires.

EdMetrics Font-Editing Options

2.0 UPDATE: Font style modifiers, like bold, and italic can now be added directly into the font command in the Edit Window. See page 18.

EdMetrics main dialog (Figure 7-2 above) contains several options for editing your font metrics. That is, you can take a “base font” and apply certain transformations to the metrics to create an altered font.

First, there are the **QuickDraw attributes** checkboxes (Bold, Italic, and so on) listed under the QuickDraw font name. These affect the drawing of the font in QuickDraw contexts; that is, when drawing to the screen or to a QuickDraw printer. (This is true even for PostScript fonts.) If an appropriate outline (TrueType) font is available, these checkboxes will select that font to be drawn and the printed results will be of good quality. If the particular transform is instead applied to a base font, the result may be a low-quality transform: for example, a “bit smear” for boldface. This is no real substitute for a true bold, italic, or bold-italic font.



AGaramond-Regular

□ Outline

□ Condense to [ ] percent

□ Slant to [ ] percent

□ Track to [ ] percent

A number of **PostScript modifiers** let you alter your PostScript fonts for printing. Condense , Slant, and Track apply the specified transforms by the specified percentages:

- Outline draw the outlined character instead of the filled character.

- Condense (or “extend”): shrink or stretch the drawn character.

- Slant slant the drawn character. (Slanted text is different from true italic, though often used for the same purpose.)

- Track: loosen or tighten the inter-letter spacing. (**Tracking** adjusts the distance between letters.)

NOTE: Unfortunately, of these PostScript transformations, only tracking is visible on the screen at this time. (This is also true of printing on QuickDraw printers.)

Add Fonts. . . Command

If you temporarily need a font that's not installed, you can use the Add FontsÉ command to add fonts or metrics files to Textures' current set. Add FontsÉ brings up a standard file dialog so you can find and open font files. Once you've added a font (and its metrics, if necessary), Textures can use it for typesetting during the current session. Fonts added in this way are *not* permanently installed, and will remain available only until you quit Textures. To install the font permanently, you'll need to quit Textures, put the font suitcase or metrics suitcase into the TeX Fonts folder, and restart Textures.

NOTE: Add FontsÉ does *not* install PostScript fonts; only font metrics or QuickDraw fonts (bitmap or TrueType).

If you have installed your fonts correctly, you shouldn't have to use the Add FontsÉ command for normal Textures operations.

This finishes our survey of how Textures cajoles TEX fonts and the Macintosh font system into working together. We'll finally introduce Textures' facility for defining customized virtual fonts.

Virtual Fonts

Textures' virtual font capability makes it possible to define custom fonts that include characters pulled from several different fonts or even "characters" that are separate pictures or graphics. A **virtual font** is a font that exists in a logical but not a physical sense. To be precise, a virtual font consists of a font metric (like other fonts) that contains a mapping table (unlike other fonts), with entries that may point to characters that physically belong to several different fonts, to user-defined characters, or even to pictures. With virtual fonts, you can, for example:

- Build virtual fonts with accented letters that are considered "real" letters by TEX, so that they can participate in hyphenation tables. This is a great advantage for non-English-language uses.*

* The usual TEX way of adding diacritical marks is to insert a special control word in your text: for example `\"u` will yield ü. When TEX is looking for points where it can break a line, it will not break words that include control sequences, so a word like *Fußgängerunterführung*, for example, will never be hyphenated.

- Substitute alternate characters, such as small capitals, old-style numerals, or “expert” ligatures, into a font.

- Build special logo or symbol fonts (or include graphics in a font).

From the standpoint of most users, a virtual font is no different from any other font in actual use. The MathTime font, for example, is a virtual font, but from the user’s standpoint, it looks just like any other font. However, a few provisos apply:

- Every font included in a virtual font must be physically present on your system in order to print.

- If characters from too many downloadable fonts are used, you are likely to run into printing problems due to printer memory constraints.

  When you use a character in a virtual font that is mapped from a downloadable font, the entire component font will be downloaded to the printer, even if the virtual font uses only one character from the component font. (This limitation is an artifact of Apple’s downloadable fonts scheme.) With the MathTime font, this is not a problem, because the Times font that MathTime utilizes is built into the LaserWriter ROM.

Defining your own virtual font is presently a somewhat involved procedure, but you don’t necessarily need to know anything about virtual fonts in order to use them. To program your own virtual font, please refer to Appendix C, “Building Virtual Fonts,” beginning on page 211.

chapter 8

Pictures

Textures integrates the Macintosh's graphic potential with TEX so that you can include pictures from many sources, scale them, and position them in your typeset documents. This chapter explains how you can incorporate both Macintosh and PostScript (EPSF) graphics into your Textures files.

CONTENTS

Pictures in Textures

As we've mentioned, TEX typesets with "boxes," without having to concern itself with what goes inside a box. Built-in extension facilities allow TEX to be customized so that these boxes can include graphics produced on various computer systems. Even though TEX itself may know only the external dimensions of a picture, Textures can supply the image for any Macintosh graphic so that you can view it in the Typeset window and print it to a printer or phototypesetter.

You can place pictures in your Textures documents via your document's Pictures window facility, or you can directly include pictures contained in external files:

- The **Pictures window** works much like the Macintosh Scrapbook, letting you store Macintosh (QuickDraw) graphics that you can then place in your document via TEX macros.

- An **encapsulated PostScript file (EPSF)** is a PostScript graphic contained in a separate file. An EPS file can (and should) also contain a Macintosh (PICT) representation of the figure so that it can display on the screen.

- **PICT files** are files containing Macintosh-format (QuickDraw) graphics.

NOTE: You can place commonly-used EPS and PICT files in the **TeX Inputs folder**, where Textures will automatically find them. Textures looks for an included graphic file first in the current directory and next in the TeX Inputs folder. You can also specify an explicit pathname for your file. Note that neither filenames *nor* pathnames (i.e., folder names) may contain spaces. (See "Where Textures Looks for an Included File" on page 68 for more information.)

There are a number of ready-made picture macros that let you conveniently place your pictures into a Textures document. In the following sections, we'll look at when you might want to use the various kinds of graphics, and survey some picture-placement macros that are available. In the next chapter, we'll look at how you can build your own macros with TEX's `\special` command.

QuickDraw (PICT) vs. PostScript (EPSF) Graphics

QuickDraw is the Macintosh's native graphics language, and was originally designed for representing images at the resolution of the Macintosh screen (72 dots per inch). Later extensions have brought higher resolution and color to QuickDraw, but at present, PostScript graphics remain the best choice for capable and portable production-quality imaging.

Higher-quality printers for the Macintosh describe the printed page in **PostScript**, an industry-standard "page-description language" used to drive output devices such as printers, color printers, and phototypesetters. PostScript images can consist of characters, shapes, and digitized images; these can be in black-and-white, gray scale, and color. PostScript is essentially device-independent and has become the most widely used standard for controlling computer printers and imagesetters.

Drawing programs such as Adobe Illustrator and Deneba Software's Canvas can produce high-quality pictures drawn in PostScript and stored in **Encapsulated PostScript Files (EPS files or EPSF graphics)**. Many scanners can also produce EPS-format files. When saving a file as an EPS file, be sure to select the option to save a Macintosh-format screen image if you want to be able to see it in your Typeset window. (But even if no screen image is present, the graphic will still print properly to PostScript output.) An EPSF graphic is a complete PostScript representation of an image that may include both graphics and text. In addition to this PostScript picture, EPS files should also contain a *screen image*, or *preview*, that represents the PostScript picture in the Macintosh's PICT (QuickDraw) format for display on the Macintosh screen. If this screen image is present, Textures will display it in the Typeset window.

EPSF graphics will produce high-quality results only on PostScript output devices, though the *preview* illustration will print on QuickDraw printers.

Including Encapsulated PostScript Files

At the most primitive level, you can include EPSF graphics in Textures documents with the TEX `\special{illustration...}`

command, specifying the name of the file that contains the encapsulated PostScript picture. However, the `\special` command by itself *does not* provide space for the figure. One of the advantages of EPS files is that they contain “bounding box” information specifying the picture’s dimensions. Several public-domain macros have been written that will access this information directly and do the job of providing space for you.

Another advantage of some ready-made macros such as the BoxedEPS macros (introduced below) is that they alert you in the TeX Log window if a requested picture file is not found (a common occurrence if you change a file’s location, say, in the process of moving it from one computer to another). By themselves, `\special` commands report *no error* if a file is not found.

Macros for Placing EPSF Graphics

The macro sets introduced in this section all place EPSF graphics in your Textures documents. In each case, the files that define these macros should be placed in the TeX Inputs folder; you can then make them available in your Textures files by including the appropriate `\input` statement.

NOTE: These macros are available free of charge, from `ftp.bluesky.com` in `/pub/graphics` (see page 224 for information on contacting Blue Sky Research). These versions are specifically adapted for use with Textures—copies transferred from a mainframe or PC will not work.

“BoxedEPS” The BoxedEPS macros, written (and well documented) by Laurent Siebenmann, automatically allocate space for EPSF graphics and provide extensive facilities for displaying and adjusting a figure’s bounding-box dimensions.

To use the BoxedEPS macros, you’ll need to include the following statements in the preamble to your Textures file:

```
\input BoxedEPS
\SetTexturesEPSFSpecial
```

(Or, include these statements as part of the format you are using.) To prevent the outlines of the bounding boxes from being displayed, you'll also need the statement

```
\HideDisplacementBoxes
```

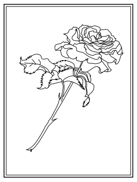
The primary form of the picture-placement command is then:

```
\BoxedEPSF{filename [scaled nnnn]}
```

This command takes the EPSF graphic contained in the file `filename` and inserts it at the current location as a TEX box of the dimensions of the EPSF bounding box. If the optional `scaled` factor is present, the figure is scaled by a factor of `nnnn`/1000 (so that "1000" = 100%). For example, the command

```
\BoxedEPSF{SmallRose scaled 400}
```

inserts the illustration SmallRose and scales it to 40% size:



The BoxedEPS macros include a number of macros for auxiliary functions associated with placing and scaling EPSF graphics. They are also fully compatible syntax-wise with the "EPSF Macros" described below, such that a TEX file written to use the EPSF Macros set will behave in the same way if the BoxedEPS macros are used instead.

"EPSF Macros" The EPSF Macros, by Tomas Rokicki, also find bounding-box information and automatically make space for your EPSF graphics. To use these macros, begin your Textures file with the statement

```
\input epsf.def
```

You can then insert EPSF graphics with the macro

```
\epsfbox{filename}
```

The EPSF Macros especially provide for compatibility with TEX systems on UNIX machines.

“PSFIG” To access the widely used PSFIG macros, by Trevor J. Darrell, begin your file with the statement

```
\input psfig
```

You can then insert pictures with a command in the form

```
\psfig{figure=filename,height=hhh,width=www}
```

Note that spaces are not allowed.

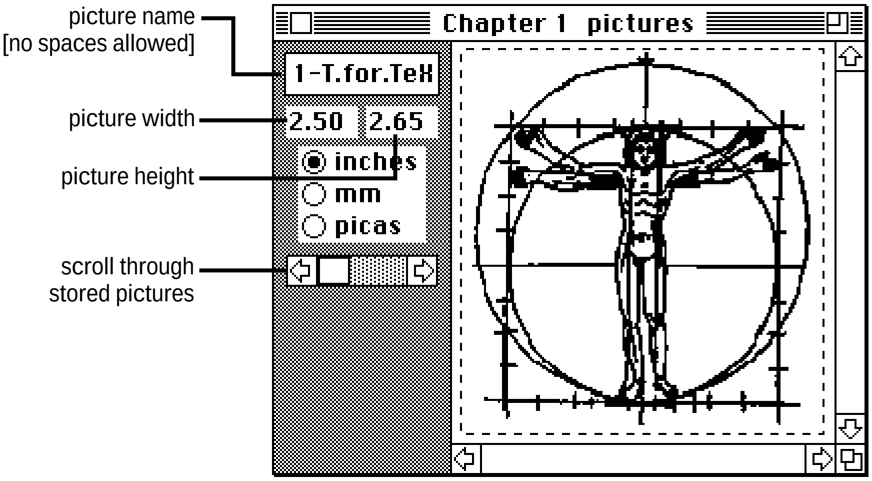
For example:

```
\psfig{figure=smallrose,height=2.5in}
```

We’ll now look at two methods of incorporating standard Macintosh graphics into your files: via the Pictures window, and via a separate PICT file.

Including “Cut-and-Paste” Pictures

A Pictures window is attached to every Textures document. The Pictures window works much like the Macintosh Scrapbook, enabling you to transfer pictures created in applications such as MacPaint to your Textures documents. The name and dimensions of the currently visible picture are displayed in the gray area on the window’s left side (Figure 8-1).



1-T.for.TeX

2.50 2.65

● inches
○ mm
○ picas

Figure 8-1. A document's Pictures window.

You can't make any changes to pictures in the Pictures window—any picture editing must be done in the application that created the picture.

Including pictures via the Pictures window is a two-stage process:

1. First, collect the pictures: Copy each picture from its original application and Paste it into your document's Pictures window.

2. You can now place these pictures in your typeset document by means of one of the picture-placement macros or by using TEX's `\special` command directly.

We'll now look at the details of this process.

Collecting Pictures

You can cut and paste graphics from any Macintosh application into a document's Pictures window via the Macintosh Clipboard or the Scrapbook desk accessory. (Review your *Macintosh User's Guide* if you are unfamiliar with how to cut and paste graphics.)

To make a document's Pictures window active, just select it (Document pictures) from the Windows menu. After you have pasted a picture, be sure to name it by typing a one-word title in the blank rectangle at the top left of the Pictures window: TEX cannot place a picture unless it is named here. No spaces are allowed in picture names.

Pictures are automatically stored in alphabetical order as you enter their names. You can flip through the pictures one at a time by clicking the scroll bar arrows. You can paste as many pictures into the Pictures window as available disk space allows. You can remove a picture with the `Edit` menu's `Clear` command.

By pasting a picture to a document's Pictures window, you actually add it to the resource fork of your Textures document file. (No explicit `Save` command is required to save it.)

NOTE: Textures can only see pictures in an input file *when the input file itself is actually open*. However, TeX's `\input` command closes an input file after reading it, so if an input file includes pictures in the Pictures window, you'll need to open the input file beforehand and *leave it open during typesetting* to make those pictures available. Textures will search any open file for pictures, and you may find it most convenient to create a separate "pictures-only" Textures file and simply leave that file open when you typeset.

Picture Dimensions

Below the picture's name in the Pictures window are two rectangles that display—on the left—the active picture's width, and—on the right—the picture's height. You can choose to display these dimensions in inches, millimeters, or picas by clicking the appropriate button. You'll need these dimensions when you place or scale a picture in your document.

NOTE: If you place your Macintosh graphics in PICT files rather than the document's Pictures window, there is no way to get these dimensions directly—you must measure the picture yourself, whether by eye, by hand, or with the magnifying glass tool.

Macros for Placing Pictures from the Pictures Window

The file **pics.tex** (for LaTeX, **picmacs.tex**) in the TeX Inputs folder contains two macros, `\picture` and `\centerpicture`, for placing pictures from the Pictures window. To use these macros, begin your TeX file with the statement

```
\input pics.tex
```

or, for LaTeX,

```
\input picmacs.tex
```

Once you've input one of the picture macros at the beginning of a file, it functions like any other TeX definition. Their syntax is as follows:

```
\picture width by height (picturename scaled nnnn)
```

and

```
\centerpicture width by height (picturename scaled nnnn)
```

where `width` and `height` are the natural dimensions of the picture, before scaling.

The picture name in these picture commands should match the name of one of the pictures pasted into the document's Pictures window. These macros require that you include the `scaled nnnn` phrase. (As with the EPSF macros, `scaled 1000` indicates a scale factor of 1.000, or no scaling.)

NOTE: If TeX can't find the picture you specify, it will *not* report an error message to the TeX Log window to tell you about it, and the document will simply be typeset without it.

You can also refer to pictures from the Pictures window directly with the `\special{picture...}` command, which is defined in Textures and described below on page 167.

chapter 9

Specials: Color, Pictures, PostScript

TEX's "special" control sequence is a facility for extending standard TEX to include implementation-specific extensions, such as placing graphics. Textures includes a number of these extensions; for example, all of the picture-placement macros we've mentioned work by means of special commands. Other special commands allow the Textures programmer to control the color of TEX's 'pen' (and thereby the color of elements of the page), and to program directly in the PostScript graphic language.

Special commands are primitive TEX commands, for use by TEX programmers rather than TEX users. See *The TEXbook*, page 228 for more information on the special command itself.

CONTENTS

Numeric Factors: Real and Integer

Some of the color and picture commands described below include numeric factors, e.g., for picture scaling and color saturation. These may be either integer or real numbers, which are interpreted in two very different ways *depending solely on the presence or absence of the decimal point.*

Integers (without a decimal point) are divided by 1000 in a fashion similar to TeX’s magnification system, so that the integer ‘`1000`’ gives an actual value of unity (1.000). A real number (with a decimal point) is interpreted as a conventional real number. This means that, e.g., the integer ‘`1`’ gives the actual value 0.001, very different from the real number ‘`1.`’ We repeat: *integer and real numbers are interpreted very differently, and are distinguished solely by the presence of a decimal point.*

Numeric factors may end with an *extra decimal point*; this is allowed as a terminator to force interpretation as a real number, as in this example:

```
\def\rgb#1#2#3{\special{color rgb #1. #2. #3.}}
```

Here the parameters to the `\rgb` macro are always interpreted as real numbers, whether or not the caller has supplied the decimal point.

(To avoid any possibility of confusion, we recommend the habit of *always* including the decimal point. The principal justification of the different interpretation of integers is that TeX macros can generate integers much more easily than real numbers.)

Color Special Commands

Textures supports an integrated color drawing model: color is handled uniformly, and appears on color monitors, on all types of color printers, and in typeset material exported in QuickDraw and PostScript (EPSF) form.

Nothing need be done to turn color “on”. The drawing color is chosen by simple `\special` commands, but since it defaults to black, you’ll see no difference until you (or macros chosen by you) select a different drawing color.

The primitive color programming commands described here are probably best used by experienced TeX programmers. Most users will prefer to copy definitions from the sample color documents, or to use the color facilities of LaTeX. (The sample file `crayola` gives a set of colors matching the 64 color Crayola crayon set.)

Set the ‘Pen’ or Drawing Color

These special commands each perform a primitive color operation: set the drawing color as specified.

```
\special{color rgb red green blue}
\special{color cmyk cyan magenta yellow black}
\special{color gray gray}
```

The `rgb` color space has primary colors of red, green, and blue; this corresponds relatively well to the human eye, and to computer monitor phosphors. The `rgb` color space is additive, so increasing the three primary colors to full saturation gives pure white of maximum intensity.

The `cmyk` color space has primaries of cyan, magenta, yellow, and black; these are the conventional ink colors of 4-color process printing. The `cmyk` colors are subtractive, so that increasing each factor darkens the result. In theory, solid black can be achieved by fully saturating all of the three ‘color’ inks, but in practice the printed results are much more stable with the addition of the fourth black ink.

The `gray` color space isn’t what most people would call a color space at all. It’s just shades of gray. (Actually shades of black, in that ‘`gray 1.0`’ means pure black.)

In each of the color coordinate spaces, the factors describe the intensity or saturation of the primary color(s) of that space; these primaries may be fully saturated (1.0) or completely absent (0.0) or any factor in between, such as 0.25 or 0.7. (These factors may also be given in integer units, on a scale from 0 to 1000.)

Color Stack Operators

These two commands manage a ‘stack’ of colors, where you can ‘push’ the current color onto the stack, and ‘pop’ the previous saved color off of the stack.

```
\special{color push}
\special{color pop}
```

This is useful, e.g., for assigning a local color to an element in a colored environment: simply ‘push’ the current color, set the spot color, position the element, and ‘pop’ to restore the previous color.

This stack of colors is maintained across page boundaries, so it’s possible to break to the next page without losing the current color context.

Define a Named Color

```
\special{color define name colorspec}
```

The ‘`define`’ color special command associates a name with a given color specification; this name may then be used in other color specifications to refer to the defined color. While this definition facility can be used as a user shorthand, it’s actually intended for a more specific purpose. Color names defined in this way will appear in Textures’ communication with other applications, who will usually use the names as ‘plate’ names identifying individual color printing plates to go onto a press. ‘Downstream’ programs (such as Adobe Separator or Aldus PrePrint) can then isolate these named colors to separate pieces of film, or convert them into the equivalent CMYK process color primaries.

Color definition special commands must appear on the first output page of the document.

Set the Drawing Color by Name

```
\special{color name screenvalue}
```

After a color name is defined, it can be used to set the current drawing color. An optional screen value factor can specify that the named color is to be reduced in intensity, or ‘screened’; on film output for printing it will be rendered via a halftone screen.

Picture Placement Special Commands

Each picture placement `\special` command consists of the word `\special` followed by a command keyword to specify the individual command. For example, the command

```
\special{illustration Samples:SmallRose scaled 0.5}
```

will insert the EPS file “SmallRose,” which is located in the Samples folder:

The lower left-hand corner of the picture will be placed at TeX’s current reference point.

Picture Placement Keywords and Syntax

The command keywords are understood by Textures directly and are listed here:

`illustration`	place an EPSF graphic
`picture`	place a picture from the Pictures window
`pictfile`	place a picture contained in a PICT file

The command syntax is as follows:

```
\special{keyword picture-name [scaled nnnn]}
\special{keyword picture-name [scaled xxx yyy]}
```

where:

- `keyword` is the name of the specific special command.
- `picture-name` is the name of the EPS or PICT file, the picture name from the Pictures window.

- `scaled` nnnn is an optional phrase that will reduce or enlarge the included graphic. The nnnn value may be integer or real; if an integer, it is interpreted as a TeX magnification, i.e., 1000 represents a magnification of 1.000 (100%), or actual size.

- `scaled` xxx yyy is an optional phrase, with two scaling factors. This allows anisotropic scaling, where the horizontal and vertical scaling are independent. (As above, the factors may be integer or real.)

For example, the command

```
\special{illustration Figure001 scaled 500}
```

includes the picture in the EPS file named Figure001, placing it so that the lower-left corner is at the current point of reference, and scales it to 50% size. All of the special commands place the picture's lower-left corner at TeX's **current position**. This means at the same position the next box would appear (the reference point that would have been present if a box of height, depth, and width zero had appeared in place of the picture command).

IMPORTANT! Because special commands are dimension-less, they place graphics as if they had no height, width, or depth. So, pictures will overlap text (which is sometimes desirable) unless you allow an appropriate space with other TeX commands.

Picture Placement Example

Here's an example: the `\picture` macro, introduced earlier, takes dimension parameters and automatically provides space. The following is an example of how to write such a macro:

```
\def\picture #1 by #2 (#3){\leavevmode
 \vbox to #2{
  \hrule width #1 height 0pt depth 0pt
  \vfill
  \special{picture #3}}}
```

(Here, the `\leavevmode` command ensures that a paragraph has begun, for use with LaTeX, and the `\hrule` command sets a zero-height invisible rule of the correct width for the picture.)

This macro, "`\picture`," takes three parameters: the picture's horizontal and vertical dimensions, and the name of the picture to be placed. This macro will place pictures from the Pictures window—to place pictures from EPS or PICT files, you would replace the `picture` keyword with `illustration` or `pictfile`, respectively.

PostScript Graphics and Special Effects

As we've seen, you can use the `\special` command to include Encapsulated PostScript illustrations; you can also include PostScript instructions directly in your Textures documents with the `\special{postscript...}` and `\special{rawpostscript...}` commands.

Programming in PostScript Much like TEX itself, PostScript is an extremely versatile, general-purpose, device-independent language, though oriented to low-level graphics rather than to high-level typography. By the same token, it is not necessarily easy to use. But some functions are relatively straightforward, and give you effects that you can achieve only in PostScript. For example, PostScript instructions allow you to do the following:

- rotate portions of the page
- specify halftone screens
- fill characters with a pattern
- outline characters with any line width

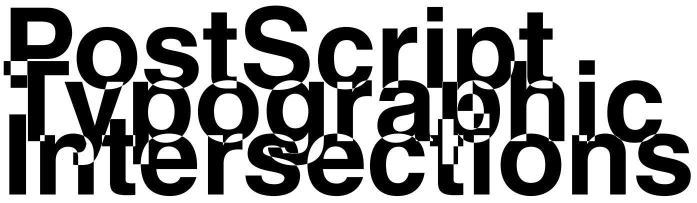


Figure 9-1. Special effects with PostScript.

Although you can include PostScript instructions directly in a Textures document, you should first gain experience with the PostScript language before trying to do so. The *PostScript Language Tutorial and Cookbook* and the *PostScript Language*

Reference Manual, written by Adobe Systems, are required reading if you want to program in PostScript.

The results of direct PostScript instructions will *not* be visible in your Typeset window: any graphics or special PostScript effects will show up only in the printed output. Also be aware that no PostScript error checking by Textures is possible—PostScript commands are passed straight through to the printer for processing there. Any errors in your PostScript instructions will produce either unpredictable output or, more likely, no output at all.

Placing PostScript Commands in a Textures File

You can insert PostScript instructions directly in a Textures file with the `\special{postscript...}` command, followed by the PostScript instructions that you want to send to the printer. A simple example of such a PostScript sequence is:

```
\special{postscript newpath 0 0 72 0 360 arc stroke}
```

This draws a one-inch circle centered on the current point.

NOTE: Like all special commands, the `\special{postscript...}` command may contain TeX macros that are expanded before transmission to the printer—TeX scans through PostScript text that appears in special commands and performs its usual macro processing. While this offers great potential to the experienced programmer, some PostScript sequences that look like TeX commands may trip TeX up. Unless you know both TeX *and* PostScript quite well, it is safer to include your commands in a separate PostScript file.

For an example, see the sample file “PostScript example.”

Including a PostScript File

You can send a *file* of PostScript instructions to the printer by using the `\special{postscriptfile...}` command followed by the name of the file. This is very similar to including an EPS file, but unlike an EPS file you cannot view the graphic on the screen. There is also a `rawpostscriptfile` keyword, parallel to the `rawpostscript` command introduced in the next section.

As with the `postscript` special command, Textures places `postscriptfile` instructions in an environment scaled so that there are 72 coordinate units to one inch and (0,0) is at the current TEX position.

PostScript Preamble Definitions

```
\special{prePostScript definitions}
\special{prePostScriptfile filename}
```

These commands, for sophisticated PostScript programming, add definitions to the PostScript dictionary used by Textures. The first form adds literal PostScript definitions; the second form adds the contents of an external file of PostScript code. These dictionary definitions may then be used by PostScript special commands within the typeset document, reducing the overhead of lengthy PostScript inclusions.

These PostScript preamble commands must appear on the first output page of the document.

Programmer's Note: The PostScript Environment

When you include PostScript instructions, you should be aware of the surrounding environment in which those instructions will be placed:

- `postscript`, `postscriptfile`: Textures sets up a PostScript environment that looks "normal." That is, Textures sets up a coordinate system scaled so that there are 72 coordinate units to one inch, and translates the coordinate system so that (0,0)—the origin of PostScript's coordinate grid—is at the current TEX position. The entire PostScript inclusion is bracketed by PostScript `save` and `restore` save-state operators, so that your PostScript instructions will not modify Textures' drawing environment.

- `rawpostscript`, `rawpostscriptfile`: These commands inserts PostScript instructions directly into Textures' output stream, letting you modify the Textures drawing environment.*

* Just to keep you working, Apple's QuickDraw and Adobe's PostScript use inverse coordinate grids. QuickDraw is screen-oriented, and the origin point is the upper-left corner of the screen. Textures' internal PostScript

It’s best to gain familiarity with Textures’ use of PostScript by examining the PostScript output from the printer driver. Textures’ internal dictionary is subject to change without notice; use these commands at your own risk!

- `illustrator`: This command inserts PostScript instructions in the Adobe Illustrator idiom directly into the Textures output stream. This command is applicable only when typeset pages are being saved in Illustrator format, via the Save AsÉ command; it is ignored otherwise. Note that other PostScript inclusions are ignored while creating Illustrator format files.

framework matches QuickDraw, so the vertical axis is inverted from the usual PostScript model.

chapter 10

Printing

The basics of printing were introduced in Chapter 2; this chapter extends the discussion with some considerations specific to various output devices.

CONTENTS

Printing Typeset Documents

Once you have typeset your TEX file, you can print it with the PrintÉ and PrintOne commands. To print typeset pages, the Typeset window must be the active (topmost) window on the screen. (In Flash mode, the Typeset window is always the window that is printed, unless you close it first.)

Printing a Textures document is like printing other Macintosh documents:

1. Select a printer with the Chooser desk accessory (in the Apple menu). If you're using the AppleTalk network to connect to a printer, make sure it is shown to be connected. (What you are actually doing in the Chooser is first, choosing a printer driver, and second, choosing a printer. Most PostScript printers use the LaserWriter driver.)

   See your *Macintosh User's Guide* if you need more information about the Chooser.

2. If you are changing printers (or printing for the first time), verify the Page SetupÉ settings in the File menu.

3. Select one of the Print commands to print the document:

   PrintOne prints only the currently active page of the document in the active window.

   PrintÉ lets you print the entire document, or any consecutive pages that you choose. A dialog will appear with the name of your printer and the version number of its printing resource at the top and containing a variety of printing options.

NOTE: Make sure your document has finished typesetting before you print: only pages that have already finished typesetting will be printed.

NOTE: In Flash mode, the Print commands will print the contents of the Typeset window of the active document, whether it's the topmost window or not. To print the contents of the Edit window, close the Typeset window or deselect Flash mode.

You can also "print" to a fax modem, to fax your Textures documents, or to a PostScript file; for example, in order to take that file to another machine and print it there, without installing Textures.

NOTE: It is the *sequential page number* of the document that selects the pages to be printed, not the TEX (or logical) page number or the page number that appears on your printed page. For example, selecting page three of the material from the PrintÉ dialog will not necessarily print the page that is labeled “Page 3.”

The following sections give some specifics for phototypesetting machines, PostScript printers, and non-PostScript (QuickDraw) printers.

Output to a Phototypesetter

We introduced the PostScript language in previous chapters; Textures generates PostScript output so that you can reproduce your document on any of the host of phototypesetting machines that now understand the PostScript language. (To be precise, these machines are photo-*image*setters, since PostScript generates graphics as easily as type.) PostScript-capable phototypesetters include machines from all of the major manufacturers such as Autologic, Compugraphic, Linotype, Varityper, etc., and many of these machines can produce output at well over 2000 dpi.

Phototypesetters produce the highest-quality output of all printing devices that can be used with Textures—for example, the Linotronic 100 has a resolution of 1270 dots per inch, while the Linotronic 300 produces output with a resolution of 2540 dots per inch. On a phototypesetter, you can use Textures’ printed output as a high-quality reproduction master for use on an offset press or other multi-copy duplication device. Phototypesetters like these don’t print on regular paper, but on photographic paper or film.

The speed at which photosetters print your document depends on whether you have inserted pictures or are using downloaded fonts, but you can expect most photosetters to require more time than Macintosh-compatible laser printers.

PostScript-capable phototypesetters show up on the Chooser associated with a LaserWriter icon.

Printing to a PostScript Printer

Like phototypesetters, PostScript printers such as the members of the Apple LaserWriter family use the PostScript page description language to produce output text and images. (That is, excepting low-end QuickDraw-only LaserWriters such as the LaserWriter II SC.) PostScript printers actually contain a dedicated computer with a PostScript interpreter built into ROM; on newer LaserWriter models, RAM can be expanded or a hard disk attached to allow for more fonts and faster printing.

NOTE: A number of printing problems can be caused by exceeding the available printer memory, usually because a document calls for a large number of downloadable fonts (such as the CM fonts). See the Electronic Help Pages on the Textures CD for more information together with a description of possible solutions.

Page Setup

The `Page SetupÉ` command brings up a dialog with the name of your printer and the version number of the printing resource at the top. With an Apple LaserWriter printer, the following options are available to you:

Smoothing Selecting `Smoothing` in the Page Setup dialog tells the LaserWriter to smooth bit images to reduce their jagged edges. (This affects only bit-mapped pictures that you have inserted into a Textures document.)

NOTE: If you're printing bit image pictures at a *scaled* size, the Smoothing option will require considerable extra time for printing.

Magnification The Page Setup dialog for LaserWriter printers lets you reduce or enlarge your document from 25 percent to 400 percent of its actual size. Enlarging your output may make it easier to proofread, and is useful for making large transparencies for presentations. Enlarged fonts can slow down the printing process.

Paper Margins No matter what page width you specify for a Textures document, your printer will determine the actual physical margins of the printed area of the page. On Apple LaserWriters, margins will usually be at least one-half inch on all sides for letter-sized paper, and at least seven-eighths of an inch on the sides and five-eighths of an inch on the top and bottom for legal-sized paper. It's a good idea to allow for more than these minimum margins. (Textures itself doesn't care—it will let you place images anywhere on or off the page.)

Downloading Fonts LaserWriters and other PostScript printers have a set of resident fonts, such as Times and Helvetica (see page 119). PostScript fonts that aren't built-in to the printer must be "**downloaded**" from the Macintosh to the printer, which slows printing. Textures automatically downloads PostScript fonts used by a document—you can also manually download fonts using programs such as Apple's LaserWriter Font utility or Adobe's PostScript Downloader program.

Each downloaded font requires printer memory. (In Apple's font downloading scheme, even if a single character from the font is used, the entire font is downloaded.) You will probably be able to download only three or four fonts at any one time with older LaserWriter models. Fonts take 20–40K of memory each, but there is no easy way to tell how much memory is available for fonts. The original LaserWriter, LaserWriter Plus, and LaserWriter II NT are not expandable and often have memory problems. Expandable printers such as the LaserWriter II NTX and II f/g and newer printers have little problem.

For more information, check the Electronic Help Pages on your Textures CD.

NOTE: The LaserWriter Page Setup option `Unlimited Downloadable Fonts in a Document` permits you to print documents containing many fonts that would otherwise generate printer errors. However, this option will cause *very* slow printing; use it only if necessary!

Printing to a QuickDraw (Non-PostScript) Printer

Most non-PostScript printers provide a lower-cost, less flexible alternative to PostScript printers. But by using the **Adobe Type Manager** (**ATM**), you can print PostScript fonts (but not PostScript graphics) on a non-PostScript printer such as the Hewlett-Packard DeskWriter, achieving quality comparable to a PostScript LaserWriter. For more information on the Adobe Type Manager, see page 131.

NOTE: If ATM is *not* installed, PostScript fonts including the Computer Modern PostScript fonts and the AMS PostScript fonts will not work properly with a non-PostScript printer.

TrueType fonts for which you have metrics will print without difficulty.

chapter 11

Connecting with Other TEX Systems

This chapter begins by comparing Textures to other TEX installations, and then discusses how to transfer files to and from TEX systems on other computers.

CONTENTS

Compatibility with Other TEX Systems

We can consider Textures' compatibility with other TEX systems under four headings: at the level of text (input) files, at the level of DVI (typeset output) files, in regard to font metrics, and in regard to fonts themselves.

Text Files Text-only files are completely compatible. Depending on the system, you may need to convert carriage returns to line-feed characters, or vice versa, but that's all. The next section lists some ways of transferring text files.

NOTE: Recall here that `\special` commands are implementation-specific, and you may need to modify the syntax of any `\special` commands that you've used. (This is another advantage of relying on well designed picture-placement macros such as the BoxedEPS macros.)

DVI Files DVI files are not normally created as separate files by Textures, but you can use them, as explained below on page 185. You can also open DVI files from other TEX systems directly with Textures. We recommend, however, that you transfer text files and re-typeset them, if possible. (Text transfer is easy, and binary transfer between different systems may be difficult.)

Font Metrics Other TEX systems store font metrics as TFM (TEX Font Metrics) files. The EdMetrics utility can import TFMs in the PL (Property List) format. (A PL is a text-file version of a TFM.) Other TEX installations should include a program called "TFtoPL," which can translate a TFM file into a PL. (Textures can also create PL and VPL files via the "Make VPL" button in EdMetrics' file dialog.)

If you transmit files that make use of virtual fonts, the other TEX system must have virtual-font capability and the proper VPLs and base fonts in order for the font to work.

Fonts Themselves "PK" is a common TEX bitmap font format. Tools for converting PK to Macintosh fonts are available free of charge from Blue Sky Research. (These tools run under the MPW Shell.) Also, a Macintosh application of METAFONT, the system for font design by Donald Knuth, is available free of charge. See page 224 for contact information.

We'll now look at how to transfer TeX text files and typeset documents to and from other computers.

Transferring TeX Text Files to and from Other Computers

Because TeX takes standard text-only (ASCII) files as input, Textures can process text files written on any other system. A number of methods are available for transferring files to and from other computers, including:

- e-mail
- network connections (remote mount)
- Apple File Exchange
- DOS Mounter

NOTE: If you e-mail a Textures file, you will send across *only* the document's text (i.e., the text fork of the file), but not its typeset pages or any pictures that may be in the Pictures window.

Now we'll look at how to transfer typeset TeX documents to and from other TeX installations via TeX's DVI page output format.

DVI Files: Importing and Exporting Typeset Documents

TeX has its own **DVI (Device Independent) file format** for representing typeset pages, and typeset pages are normally "shipped out" to a DVI file as they are typeset. DVI files are written in compact binary code, and are not human-readable. In Textures, a document's DVI "file" is not actually a separate file, but is stored in the resource fork of the same file that contains the source text.

There is no single standard tool or method for transferring binary files, and transferring DVI files between computers may present a variety of difficulties. Transferring text files is generally the most problem-free option.

Saving a Typeset Document as a Separate DVI File

In transferring DVI files to other systems, you can create DVI files via Textures Save AsÉ command. With the Typeset window as the active window, simply select the `DVI` option to the `Save AsÉ` command. (You will then need to accomplish the binary file transfer to the other computer.)

Importing DVI Files: Using Textures as a DVI Reader

Textures will directly open DVI files from other systems. Imported DVI files may be previewed and printed like Typeset pages from Textures documents.

NOTE: Due to differences in graphics handling between platforms, graphics included in DVI files from other systems will not appear in DVI files viewed by Textures.

DVI files cannot be edited. If you wish to edit a document from another TEX system, you will need to transfer the text file, edit and retypeset on your machine.

chapter 12

Tips and Troubleshooting

This chapter includes various suggestions for getting the most out of your Textures system and for dealing with specific problems you might encounter. After looking at error processing and the operation of the TEX Log window, we'll summarize some of the most common difficulties in getting Textures to work with your system and suggest some solutions.

While you may find this chapter useful, more current and detailed information is available to help you quickly resolve problems with your Textures installation. Standard Textures installations include linked technical support help pages to guide you in the Textures Documentation Folder. For the most up-to-date information, check our web site, at `www.bluesky.com/help` .

If you cannot quickly resolve your problem with these tools, then contact us—for information on contacting Blue Sky Research Technical Support, see page 224.

CONTENTS

Tips for Working in Flash Mode

We've found the following points useful when working in Flash mode:

- Try placing a smaller Edit window directly on top of a full-screen Typeset window.

- To take best advantage of Flash mode with large documents, you can select a piece of text, copy it, open a new "working window," and paste. (You should place your document's "preamble" information into your own format to make this most convenient; see pages 103 and 111.)

NOTE: LATEX users can save processing time by building the document style into a special LATEX format:

1. Turn off Flash mode, then create a new two-line document specifying and dumping the document style—for instance,

```
\documentclass[12pt]{book}
\dump
```

2. Select LaTeX in the Typeset menu and typeset the file to make a new canned format.

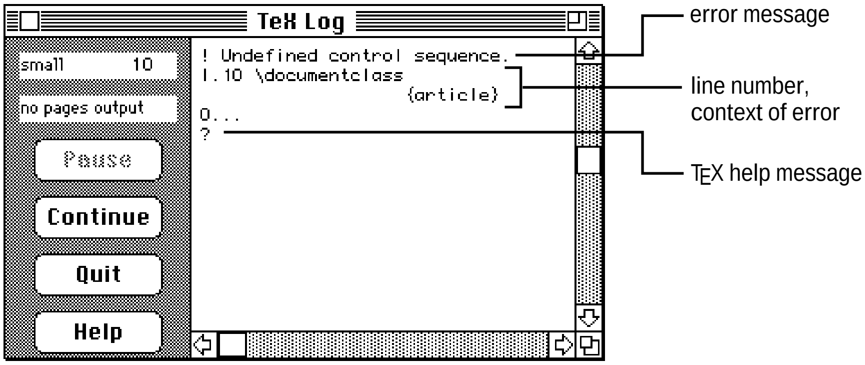
3. Save the new format in your TeX Formats folder, giving it a descriptive name, e.g. `book12`.

Any documents you typeset using the new format won't require the `\documentclass` declaration. (Refer to the LaTeX Notes File in the LaTeX Support Folder for more information.)

Error Handling: the TEX Log Window

We introduced the TEX Log window in Chapter 2, but deferred a full discussion of error messages and of the options for processing errors. We'll now look at TEX's error processing in more detail.

Reading Error Messages Error messages in the TEX log let you find and correct certain kinds of errors before the typesetting process is completed. TEX begins its error messages with `!` and displays two lines of context to show what part of the text it was reading when it detected the error, as shown in Figure 12-1.



```
! Undefined control sequence.
l.10 \documentclass
                   {article}
0...
?
```

error message

line number,
context of error

TEX help message

Figure 12-1. A TEX error message.

The top line (`l.10...`) shows what TEX has read so far, and where the error was found: here, `l.10` refers to line number 10 of page 1. The next line shows what is still to be read. (In this case, we've typeset a sample LATEX file using the Plain format, so that `\documentclass` is interpreted as an unknown control sequence.)

The `?` prompt means that TEX wants to know what to do next and is waiting for you to respond. `Pause` will be dimmed, and you can select the `Continue`, `Quit`, and `Help` buttons.

Options Available in Pause Mode

When TEX pauses for an error, the following standard TEX options are available (type the letter and a carriage return in response to the question mark prompt):

s **Scroll** future error messages, run without stopping. (Flash mode does this automatically.)

r **Run**: Like `s` but stronger (don't stop even if an included file can't be found).

NOTE: In a "traditional" (batch-oriented) TEX installation, `q` (Quiet) tells TEX to suppress error messages to the monitor and redirect error output to a separate log file. In Textures, however, all output goes to the Log window rather than to a file, and there is no difference between `q` and `s`.

1 Typing a number *n* skips over the following *n* tokens (characters or control sequences) from the input stream.

- x **Exit**: Terminate typesetting without completing the current page (the Log window remains the active window).

- e **Edit**: Return to the Edit window at the point in the file where the error was encountered. (This is the same as clicking the `Quit` button.) You can then enter the correction and re-typeset the document.

- i `text` **Insert text** before continuing to process (type a carriage return after the inserted text.) . This is for correcting errors when they are flagged and will *not* correct the original error in the Edit window.

To ignore an error and continue typesetting, type a carriage return only.

Correcting Errors in the Log Window

If you select "Help," TeX generates a message explaining what it thinks the trouble is and what you can do to recover. If you've misspelled a command, for example, TeX reads it as an undefined control sequence, and the message will begin:

```
The control sequence at the end of the top line
of your error message was never \def'ed.
```

To provisionally correct the error and allow TeX to continue typesetting the document, type `i` to tell TeX you want to insert a correction. Then, without inserting a space, retype the correct control sequence. Now type a carriage return or click the `Continue` button to resume.

IMPORTANT! *Even though you correct an error in the TeX log, it will remain in your text file until you go back to the Edit window and correct it there also. (It's usually easier to simply use the Quit button (or type "*`e`*"), which takes you to the source of the problem, where you can permanently correct it before retypesetting.)*

There is one circumstance in which you *must* type a correction in the TeX log before you can proceed. If your document lacks an `\end` or `\bye` statement, you will get the error message

```
(Please type a command or say \end)
*
```

You should type `\end` (not preceded by an `i`) in the Log window to properly close the typesetting job. (This is not a problem in Flash mode, which automatically inserts a `\end` statement if it is missing.)

If you're still not sure how to correct the error, you can ask TeX to ignore the next 1 through 99 characters of output by typing the number you want to skip. Then click "Continue" or type a carriage return.

For more discussion of error messages, see *The TeXbook*, pages 30–34 and Chapter 27.

NOTE: You can also have your Textures files send your own messages to the Log window on typesetting, by using TeX's `\message` command.

Problems with Typesetting

Can't typeset/can't exit from Textures

If you receive the message,

```
Please finish typesetting before quitting
```

and cannot typeset or exit, the situation is that the TeX log has paused for an error. You need to exit from the TeX log (by pressing the `Quit` button) or, in the case of a missing `\end` statement, by typing `\end`, as discussed in the section immediately preceding. To permanently correct this error you'll need to include an `\end` or `\bye` in your Edit window; but to let TeX wrap up its typesetting, you first need to include an `\end` statement in the *Log window*.

Problems with Formats

"Unknown Format"

If you get the message

```
Incompatible format file: please \dump it again.
```

this means the format file was created with a different version of Textures. You'll need to recompile your formats under the version of Textures that you are using.

To recompile a format such as LaTeX or AMS-TeX, you'll need the following files:

- The format's source (text) file or files.
- The **hyphen.tex** file for American English hyphenation, or the file with the appropriate hyphenation patterns for the language you are using.
- The VirTeX format (needed for recompiling Plain, LaTeX, or for any other format that redefines any of the Plain control sequences).

NOTE: Capitalization is not important in filenames, and the .tex suffix may be omitted.

For LaTeX: First things first. Most of the time you can avoid recompiling LaTeX altogether by simply downloading the most recent LaTeX format from our web site. Check our web pages at www.bluesky.com, or go directly to our ftp server, ftp.bluesky.com/pub/.

If you want to build a custom LaTeX format, or otherwise choose to rebuild LaTeX, refer to the LaTeX Notes file in the LaTeX Support Folder.

Format name is grayed out If a format appears grayed out on the Typeset menu it is because you haven't installed the format (by placing the format file in the TeX Formats folder). See page 107 for information on installing formats.

Problems with the Macros Menu

Macros menu is grayed out If the Macros menu appears grayed out and the Macros menu's Edit menu item doesn't function, the situation is that the format associated with the document in the active window is not installed. Macros menus are stored as part of a format file, and you can't add a menu to a non-existent (or non-installed) format.

To solve the problem, check to see that the proper format for your document is installed. To temporarily install the format, use the `Add FormatsÉ` command. To permanently install it you'll need to move the "canned" format file into the TeX Formats folder (page 107).

If the Macros Menu control(⊞) in the sidebar of the Edit window is grayed out, it means that the document's format has not (yet) been programmed to have Macros Menu. See page 92 for instructions on adding Macros to your format.

Other Problems with Textures Use and Installation

Refer to Installed Electronic Help Pages

The standard Textures installation includes a set of electronic Help Pages designed to help you identify and resolve an array of common problems encountered by customers. Subjects include:

- Problems with included graphics
- Printing problems
- Screen display problems
- Font Problems
- Using and Installing LaTeX

These pages are found in the Technical Web Pages folder in the Textures Documentation folder in the Textures folder.

Use the Internet for Most Current Technical Support

The Electronic Help pages included in your installation are a dated subset of our on-line technical support system. If you don't find the solution there, get the most current Textures Technical Support information on our web pages, at`www.bluesky.com/help` .

appendix

A

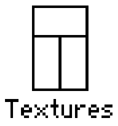
Textures Files and Folders

This Appendix gives an outline listing of your Textures distribution files.

NOTE: Due to publication schedules, the information here may be superseded by information in the *Textures Installation Guide*; refer to your *Installation Guide* for further specifics.

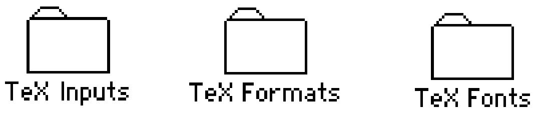
Files that You Must Have to Run Textures

Not counting font files or LATEX, your Textures package includes more than 40 files. (LATEX adds another 300 or so to the list.) Textures' Installer program automatically installs all of these files (including font files). If disk space is a constraint, you may want to know which parts of the Textures tree you can prune off without dire consequences. To switch the metaphor, we can look at the Textures system as consisting of several layers.

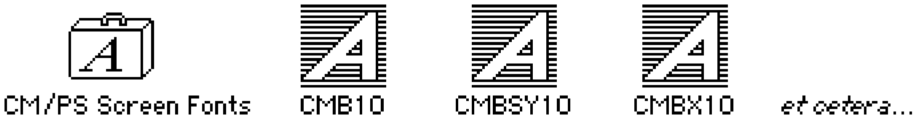


At the most fundamental level, the **Textures application file** is all you need. That is, if you work with the standard PostScript system fonts, such as Times and Helvetica, the Textures file alone will let you successfully run Plain TEX (no formats, no CM fonts).*

You can then add layers:



To read **input files** or install your own **formats** (both of which you will probably want to do), or to use **fonts** whose metrics aren't built into Textures, you will need these three folders. These folders can initially be empty.



CM Fonts: By default, Plain TEX uses 16 of the Computer Modern (CM) fonts. (You can, of course, override these defaults and substitute, say, Times; see page 125.) And because CM is TEX's customary font family, you will also need a fairly full set of CM fonts to happily print files you get from other TEX systems (including some of our sample files).

[*] If you need a full set of math characters but want a lean font set, you can use the MathTime fonts with Times; see page 139.

LaTeX: Your LaTeX distribution by itself consists of more than 300 files (not including the numerous font files required by LaTeX). If you don't use LaTeX, all of the various LaTeX-related files and fonts do not need to be installed from the Textures CD-ROM. If you do use LaTeX, here is what you will minimally need:

- LaTeX itself (in the TeX Formats folder).
- LaTeX document class and package files, in the `¥LaTeX` folder inside the `TeX Inputs` folder.
- LaTeX sources, in the LaTeX Sources folder. (These may be retrieved from the CD-ROM if you choose to build a custom LaTeX format, or need to rebuild LaTeX.)
- Auxiliary files such as specialty macro packages, LaTeX tools and other resources from the Comprehensive TeX Archive (CTAN), depending on your particular requirements. This files are not installed by the Installer, but are stored in the CD Extras folder on the Textures CD-ROM.

Sample files: these files are intended for learning purposes or to be adapted to your own uses.

The various other pieces of the system that we haven't mentioned here perform various special-purpose functions. For a brief indication of when and why you might need them, review the rest of this Appendix and the *Textures Installation Guide*.

We'll now outline the full set of Textures files.

TeX Inputs Folder

It is not necessary to include the filename extension ".tex".The TeX Inputs folder contains public-domain macro sets. These can also serve as examples or starting points for writing your own sophisticated TeX macros.

`Option_keys` Macintosh Option-key definitions.

`Pics.tex` Macros for placing pictures from the Pictures window. (See Chapter 8, page 160.)

`Picmacs.tex` LaTeX version of the pics.tex macros.

`EPlain Macros` A set of macros written by Karl Berry to extend the Plain TeX definitions for two-column formats, footnotes, fractions, commands for obeying spaces and blank lines, and much more.

`Midnight Macros` Macros written by Marcel van der Goot for: making booklets, printing double pages, and printing outlines and crop marks; vertically aligning words in consecutive sentences; a simple looping construct (meta-macros); separating arguments in macros by newlines and by empty lines; and printing address labels and bulk letters.

`Manmac.tex` The complete set of macros written by Donald Knuth for producing *The TeXbook*.

`¥LaTeX` (folder) Contains all LaTeX input files.

`¥LaTeX2.09` (folder) Contains all LaTeX 2.09 input files.

TeX Formats Folder

The TeX Formats folder is where you place "canned" format files to make them available to Textures (see page 105).

`LaTeX` LaTeX's format file (compiled and ready to use; the source files can be found in the LaTeX Sources folder).

`VirTeX` "Virgin" TeX format file. This format has no preloaded definitions or hyphenation patterns; it's used as a base for building other formats "from scratch."

TeX Fonts Folder

Metrics for the standard set of Macintosh–LaserWriter fonts and for all of the CM fonts are built into the Textures application file itself (see page 139). The TeX Fonts folder contains Textures font metrics suitcase files (🧳) for additional fonts to be used by Textures.

On initial installation, if LaTeX is installed, the TeX Fonts folder contains a file called `basic psfonts/adobe metrics` that is used only by LaTeX. These extended and virtual font metrics let you use the standard LaserWriter PostScript fonts with LaTeX.

LaTeX Support Folder

The LaTeX Support Folder contains a variety of sample LaTeX documents, including:

`Sample`	General overview of LaTeX.
`Letter`	Business letter format.
`Small`	Simple LaTeX example.
`The Frog King`	Integrated demonstration of the LaTeX book format (including the use of the BibTeX and MakeIndex tools).

Tools Folder

The Tools folder contains subsidiary tools that are used with Textures to perform special functions.

Excalibur Folder Contains the Excalibur spelling checker for TeX and LaTeX, courtesy of Rick Zaccone and Rob Gottshall. This file is stored as a self-expanding archive; just start it to uncompress it into separate files. Documentation is provided.

Samples Folder

The Samples folder contains sample files for Plain and LaTeX as well as sample Macros menus you can install. Several of our Plain TeX examples are drawn directly from *The TeXbook*; page references are given. The contents of this folder change as we get more samples; here's what's likely to be there:

`ASCII Chart` ASCII wall chart produced with TeX.

`Circle` Text laid out in the form of a circle.

`Commute` Mathematical typesetting with Computer Modern fonts; from *The TeXbook*, page 182.

`Cropmarks` Macros to produce cropmarks for page boundaries. (Used by several other examples.)

`Dropshadow` Outlined text with a shadowed box; adapted from *The TeXbook*, page 223.

`EPSF Example` Methods of incorporating an EPSF graphic into a Textures document.

`Flowers` Two-column page with cutout illustration.

`Fool` A simple example.

`Kathy's Songs` Songbook with table of contents.

`MyWatch` Fancy typography.

`Picture Example` Placing pictures from the Pictures window; see Chapter 8, page 160.

`PostScript Example` Placing PostScript commands within a Textures file; see Chapter 8, page 169. (This illustration will not preview on the screen.)

`Primes` Demonstration of TeX as a programming language; from *The TeXbook*, page 218.

`Sample Macros Menus` Sample Macros menus for Plain and LaTeX; see Chapter 4 of this manual, pages 91 to 98.

`Short Story` Sample file from *The TeXbook*, page 24.

`SmallRose` Sample EPSF illustration (see Chapter 8, page 155).

Font Tools Folder

`EdMetrics` The EdMetrics application, used to install metrics for new PostScript or TrueType fonts (see page 143).

`Virtual Font Samples` Sample virtual font constructions, including VPL definition files; also includes **Knuth VF Paper**, the technical description of VF format.

appendix B

Font Mapping

On the output side, Textures' EdMetrics tool provides fully general mapping and encoding capabilities: you can specify any font mapping and any PostScript encoding, to make Textures correspond to any font. (On the input side, what you type on the keyboard goes directly to TEX without any mapping.) We introduced the use of EdMetrics in Chapter 7 (page 143); this appendix extends that discussion by looking at font mapping in more detail.

Font Mapping

By **font mapping**, we mean mapping from the character codes (i.e., positions in a font table) that are used internally by TEX, to the “outer worlds” of Macintosh (QuickDraw) and PostScript fonts, in order to display or print the proper characters. Most standard characters are in the same positions in all three kinds of fonts, but many special characters and ligatures require mapping to appear correctly on the screen and when printed.

In TEX itself, various conventional layouts are available: “Roman” (standard text), “Italic,” “Typewriter,” and math layouts. (See *The TEXbook*, Appendix F.) QuickDraw and PostScript each have a standard character layout for text fonts, as shown in Figures B-1 and B-2 below. (Of course, none of these standards are identical.)

TEX and PostScript character layouts are flexible and can be redefined, while QuickDraw (and TrueType) layouts are fixed. The Textures font metrics for each font contain a map that transforms the TEX character layout, or “view,” to the corresponding QuickDraw font view; this map is used for drawing to the screen or printing to QuickDraw printers. For PostScript outline fonts, a second map is used to transform character codes from the TEX view to the character positions in the PostScript font, for printing to PostScript printers. PostScript fonts also allow an optional “encoding vector” that describes the positions of each character *by name*, allowing any character to be assigned to any character code.

EdMetrics Font Mapping Options

The separate EdMetrics application provides the tools to examine and modify font maps in Textures font metrics. The font mapping options provided in EdMetrics metrics-mapping dialog are described below.

NOTE: The PostScript encoding vector should correspond to the setting you specify for the PostScript map.

QuickDraw Maps

- `None` Use TEX’s character codes without modification.
- `Roman->Macintosh8bit` Maps TEX’s Roman view to the Macintosh standard view; the upper 128 characters follow the Macintosh keyboard layout. This map is recommended for most fonts.

- `Roman->Lucida` A map developed for the Lucida font family; also used for the Computer Modern and AMS PostScript fonts. This is a 128 character map that maps characters 0–32 from the TEX Roman view to QuickDraw character codes above 128, to avoid QuickDraw problems in the lower range of codes. (Not for general use.)

- `Roman->Macintosh` Maps TEX's Roman view to the Macintosh standard *for the lower 128 characters only*, while the upper 128 characters follow the Adobe standard; obsolete.

- `Typewriter->Macintosh` Maps TEX's Typewriter view to the Macintosh standard view.

PostScript Maps

- `None` Use TEX's character codes without modification.

- `Roman->AdobeStandard` Maps TEX's Roman view to the Adobe Standard layout; recommended for most fonts, particularly to be compatible with other non-Macintosh TEX systems.

- `Roman->MacintoshStandard` Maps TEX's Roman view to the Macintosh character layout. This map allows the direct use of Macintosh special characters, and is recommended for most fonts; it requires that the PostScript encoding vector be set appropriately.

- `Roman->Lucida` This map is used for the Lucida font family, and the Computer Modern and AMS PostScript fonts; not for general use.

- `Typewriter->AdobeStandard` Maps from TEX's Typewriter view to the Adobe standard layout.

A Sample Mapping

To get a view of how character mapping works, we might follow the path of a single character as it is drawn. For example, the right double-quote is character code 34 (octal 42) in TEX's view.* When Textures goes to draw

* Actually this character is usually not typed directly, but instead entered as two right single quotes which the ligature program converts to the single character code 34!

that character, it will map it through either the QuickDraw or PostScript map for that font:

- For **QuickDraw** (for example, in drawing to the screen or to a QuickDraw printer), it goes through a **QuickDraw map** to map it to character 211 (octal 323), giving the proper printer's quotes (instead of the upright double-quote at position 34 in the font).

- For **PostScript** (for example, in drawing to a PostScript printer), the situation is similar, but a bit more complicated. First, it goes through a **PostScript map** to take character 34 and map it to the equivalent character 186 (octal 272) in the PostScript map. The AdobeStandardEncoding vector connects character code 186 to the character named 'quotedblright', and gives the character name to the PostScript font machinery to draw that character.

	´0	´1	´2	´3	´4	´5	´6	´7	
´04x		!	"	#	$	%	&	’	″2x
´05x	(	)	*	+	,	-	.	/	
´06x	0	1	2	3	4	5	6	7	″3x
´07x	8	9	:	;	<	=	>	?	
´10x	@	A	B	C	D	E	F	G	″4x
´11x	H	I	J	K	L	M	N	O	
´12x	P	Q	R	S	T	U	V	W	″5x
´13x	X	Y	Z	[	\	]	^	_	
´14x	‘	a	b	c	d	e	f	g	″6x
´15x	h	i	j	k	l	m	n	o	
´16x	p	q	r	s	t	u	v	w	″7x
´17x	x	y	z	{	|	}	~		
´20x									″8x
´21x									
´22x									″9x
´23x									
´24x		¡	¢	£	⁄	¥	ƒ	§	″Ax
´25x	¤	'	“	«	‹	›	ﬁ	ﬂ	
´26x	°	–	†	‡	·	µ	¶	•	″Bx
´27x	‚	„	”	»	…	‰	¾	¿	
´30x	À	`	´	ˆ	˜	¯	˘	˙	″Cx
´31x	¨	É	˚	¸	Ì	˝	˛	ˇ	
´32x	—	Ñ	Ò	Ó	Ô	Õ	Ö	×	″Dx
´33x	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	
´34x	à	Æ	â	ª	ä	å	æ	ç	″Ex
´35x	Ł	Ø	Œ	º	ì	í	î	ï	
´36x	ð	æ	ò	ó	ô	ı	ö	÷	″Fx
´37x	ł	ø	œ	ß	ü	ý	þ	ÿ	
	″8	″9	″A	″B	″C	″D	″E	″F	

Figure B-1. Macintosh Standard character layout.

	´0	´1	´2	´3	´4	´5	´6	´7	
´04x		!	"	#	$	%	&	’	″2x
´05x	(	)	*	+	,	-	.	/	
´06x	0	1	2	3	4	5	6	7	″3x
´07x	8	9	:	;	<	=	>	?	
´10x	@	A	B	C	D	E	F	G	″4x
´11x	H	I	J	K	L	M	N	O	
´12x	P	Q	R	S	T	U	V	W	″5x
´13x	X	Y	Z	[	\	]	^	_	
´14x	‘	a	b	c	d	e	f	g	″6x
´15x	h	i	j	k	l	m	n	o	
´16x	p	q	r	s	t	u	v	w	″7x
´17x	x	y	z	{	|	}	~		
´24x		¡	¢	£	⁄	¥	ƒ	§	″Ax
´25x	¤	'	“	«	‹	›	ﬁ	ﬂ	
´26x		–	†	‡	·		¶	•	″Bx
´27x	‚	„	”	»	…	‰		¿	
´30x		`	´	ˆ	˜	¯	˘	˙	″Cx
´31x	¨		˚	¸		˝	˛	ˇ	
´32x	—								″Dx
´33x									
´34x		Æ		ª					″Ex
´35x	Ł	Ø	Œ	º					
´36x		æ				ı			″Fx
´37x	ł	ø	œ	ß					
	″8	″9	″A	″B	″C	″D	″E	″F	

Figure B-2. Adobe Standard character layout.

PostScript Encoding

It is the PostScript encoding vector that actually defines PostScript’s view of a font. To install a standard text font (i.e., not a symbol or mathematical font) such as Adobe Garamond, three encodings are provided by EdMetrics:

- `Adobe Standard Encoding` is the encoding to select for best compatibility with other TEX systems. In this case, you would need to use the standard TEX style for composite characters (that is, you can’t use Macintosh-style Option-key characters).

If you select the `Roman->AdobeStandard` mapping, Adobe Standard Encoding is the PostScript encoding to use.

- `Macintosh encoding` is the encoding to select if you want to use the full Macintosh keyboard character set: with this encoding you can directly type Macintosh Option-key characters. If you select the `Roman->Macintosh` mapping, this is the PostScript encoding to use.
- `None` means to use the encoding vector supplied within the font, i.e., no change to the default encoding. This is appropriate for symbol and other special fonts.

NOTE: There is also another level of PostScript mapping: from character metrics to the TEX view in the first place, which happens during the Adobe Font Metric (AFM) conversion to a Textures metric. An AFM file defines metrics by character name and character code (assuming Adobe Standard Encoding). With TEX, we usually want this to map on input to the TEX view of a font. EdMetrics does this internally, and an expert user can get at this AFM mapping and modify it. Contact Blue Sky Research for more information if necessary.

appendix C

Building Virtual Fonts

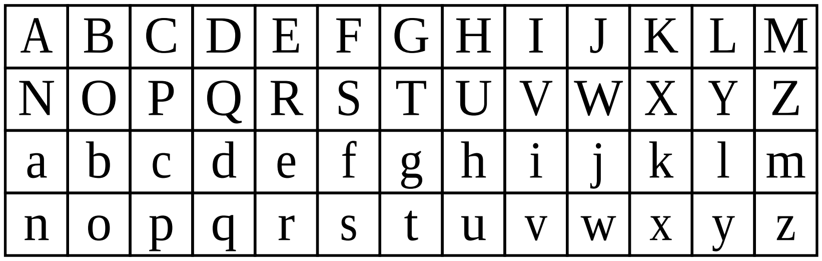
This appendix explains how to define your own virtual fonts for use with Textures. It assumes that you already have a good working knowledge of TEX's font machinery, including an understanding of AFM, PL, and VPL structures.

NOTE: You can use virtual fonts without knowing how to write one. At present, it requires some dedication to create your own virtual fonts. Since someone else may already have produced a particular font, for instance, an accented font, that you might need, you may want to check the archive sites (see page 221) for virtual fonts work that has already been done.

About Virtual Fonts

A virtual font is a synthetic font, a framework for putting together pieces of different fonts into a single, composite font. The font is called *virtual* because it exists in a logical but not a physical sense: that is, a *map* of the font exists, but the characters of the font are actually stored in various different font files.

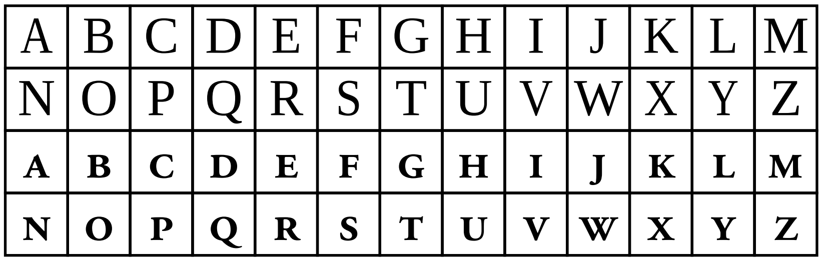
To get an overview of what a virtual font is, imagine a font table with only 52 available slots, including only upper- and lower-case letters. Figure C-1 shows the Adobe Garamond Semi-Bold font as an example:



A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
a	b	c	d	e	f	g	h	i	j	k	l	m
n	o	p	q	r	s	t	u	v	w	x	y	z

Figure C-1. Portion of Font Table (for Adobe Garamond Semi-Bold).

A virtual font provides the same frame from the TEX point of view, but may include characters from a number of various fonts or other sources. This is schematically represented in Figure C-2, where we've replaced the lower-case letters with the small-cap letters from the Adobe Garamond Expert Semi-Bold font.



A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
N	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**

Figure C-2. Multiple fonts mapped into a single virtual font.

From the TEX user's point of view, this will appear as a perfectly normal "caps and small caps" text font, with nothing to indicate that it's made up from different "real" fonts. Another use for a virtual font might be to incorporate characters from an expert font collection into a regular font so the expert ligatures are automatically available during typesetting—just as they are when you use Computer Modern. Or, you could substitute "lining" numerals for the regular digits. Virtual fonts provide almost unlimited flexibility, since you are not limited to simply moving characters positionally within a font. You can resize characters, combine multiple characters to make a new single character, add rules of any weight. You can place artwork where a character is usually found: for example, you can scan a graphic and save it as an Encapsulated PostScript file, and then put the EPS "character" into a font. Or you might decide that for each character drawn you will insert some special PostScript code to draw the characters in 50% gray rather than the usual black.

TEX's View of a Font

To understand virtual fonts, you need first to understand TEX's view of a font. As we saw in Chapter 6, the only information TEX needs to set type correctly is the metrics information for each font you use. A TEX Font Metrics (TFM) file contains the height, width, and depth information about characters in the font in a highly compact binary form. The "human readable" form of a TFM file is known as a **Property List** (**PL**), written in a PL command language created by Donald Knuth. (On other TEX systems, you can create a PL with the standard "TFMtoPL" program; in Textures, you can make a PL from a font metrics file with EdMetrics.)

A PL file contains the following information:

- comments that identify the font
- a font dimension table
- a ligature/kern table
- the metric height, width, and depth information for characters present in the font.

A PL is limited to, at most, 256 characters per font, and these characters must all be from the same "real" font (that is, from a single bitmap or outline font file).

The framework of a virtual font is a **Virtual Property List** (**VPL**). A VPL contains much the same sequence of entries as a PL file: comments, font dimension table, ligature/kern table, and metric information for each character. But, while a virtual font is still limited to a maximum of 256 characters per font, the characters can come from multiple "real" fonts. Therefore a VPL usually contains an additional set of information, called a `MAPTABLE`, identifying the fonts we're combining to make our virtual font. (*"Usually,"* because a virtual font isn't limited to ordinary characters drawn from another font.)

Creating a Virtual Font

You define virtual fonts by means of a set of **VPL commands** that extend the standard PL commands. For Donald Knuth's full description of these commands, see the file "Knuth VF Paper," (provided in the Documents folder). For more details and sample files, see "Virtual Fonts Examples" and "Virtual_Font_Construction" in the Samples folder.

The specific steps in the process for defining your own virtual fonts are given below. For more details and examples, look at the files Virtual Fonts Examples, Knuth VF Paper, and the Virtual_Font_Construction supplied in the Samples folder.

1 Create a PL (Property List) file for each of the source fonts you will use to build your new virtual font. (These are the human-readable forms of the TEX font metrics files.) You can use EdMetrics to create a PL (or VPL) from an AFM (Adobe Font Metrics) file, or you can write one from scratch in the Textures editor.

2 Create a new text file (or easier, rename one of the PL files) for use as the basefont **VPL file** and open it with Textures.

3 Insert PL and VPL commands, font information, and character information (including ligature and kerning information) into your VPL file. This will need to include the following:

A `FONTDIMEN` information, if it isn't already there.

B `MAPFONT` information, if the VPL represents a blending of two or more fonts.

4 For each new character that you want to add:

A Assign a character position for the new character.

B Insert the character's width, height and depth information.

C Insert the character's ligature and kerning information.

Be sure to remove any characters you no longer want in the font and delete character ligature and kerning information for each character that has been removed.

5 Close the VPL file.

Now you are ready to install your virtual font definition (VPL) as a Textures font:

1 With EdMetrics, open an existing metrics suitcase or create a new metrics suitcase. (This is where you will install the VPL metric.)

2 Select the `Add PL metricÉ` item from EdMetric's `File` menu, and select the VPL you just created.

3 If necessary, modify the TEX font name (i.e. the `EditingTeX font` field in EdMetrics.)

4 Click the `Okay` button and quit EdMetrics.

5 Place the metrics suitcase in the TeX Fonts folder.

You can now use your new virtual font with Textures. The next time you launch Textures, the name of your virtual font will appear in Textures' Show Fonts listing, and you can specify the font in your TEX files just as you would specify any other font, with TEX's `\font` command.

Special Effects with Virtual Fonts

In addition to combining sets of ready-made characters from different fonts to create a new font, there are other things you can do with virtual fonts.

For instance, you can create a virtual font that includes pictures and Encapsulated PostScript (EPS) graphics. An important advantage to putting often-used pictures into a font is that TEX understands the height and width information applied to a character and will be able to automatically scale and center it.

A VPL created specifically for the EPS file SmallRose is included in our sample files, along with the SmallRose file and the finished SmallRose Metrics suitcase.

In this example, we first created the basic framework of our VPL file using commands from our VPL glossary.

The next step was to determine the size of the graphic by opening the SmallRose file with Textures and finding the commented `BoundingBox` information at the beginning of the file. The size is reported in points. We get the horizontal dimension by subtracting the first number from the third, the vertical dimension by subtracting the second number from the fourth.

Once we have those dimensions, we divide by ten and transfer the information to the character position we have reserved for the EPS graphic in the VPL file. In this example, we have placed the graphic in `(CHARACTER O 46)`. We specify its width as `(CHARWD R 13.1)` and its height as `(CHARHT R 17.2)`.

We finish the character information by mapping the `(SPECIAL illustration smallrose)` into that position and closing the parentheses.

All that remains is to calculate and enter the `XHEIGHT` in the `FONTDIMEN` area of the VPL and create a metrics suitcase for the new font with the EdMetrics utility.

IMPORTANT! Since the mechanism for including pictures in a font relies on the implementation-specific `\special` *command, a VPL using such characters must be modified if you move it to another platform.*

PostScript effects such as gray, color, and so on could equally well be added.

An Example: Creating a Times "Small-Caps" Font

Modifying the Times font to create a small-caps font is a good example of a task that becomes relatively straightforward with virtual fonts:

1 You'll need a .VPL (Virtual Property List) text file containing the metrics for Times. To do this with Textures, use EdMetrics to open the Textures application file itself (Times is a built-in metric), select the Times metrics, and click the `Make VPL` button. (Note that VPL files are a superset of PL files.)

2 Edit the resulting Times.VPL file as detailed below, calling the result by some other name, say, TimesSC.VPL.

3 Convert the TimesSC.VPL text file into binary TeX metrics: again, with EdMetrics, select a metric suitcase or create a new one, and choose the `Add PL/VPL metric` menu item.

Now the font TimesSC should be available like any other.

NOTE: In a "vanilla" TeX system, the TFtoPL tool is used to convert the base Times TFM metric to a PL file, and the VPtoVF tool will convert the new VPL file back into a TFM metric and VF virtual font descriptor.

Step (2) above is the interesting part of the process, and we'll detail the sub-steps here:

A You'll need to specify *two* fonts within the virtual font. The first is regular Times, and the second is a smaller version of Times, with a scale factor that you deem appropriate, probably one that will cause the small caps to match the ex-height of the original lower case letters. (Here, we'll use 80 percent, so the small caps will be 80% of the capitals.) Font 0 is the default font, from which all the normal characters will come; font 1 is the smaller font from which we'll take our small caps.

```
(MAPFONT D 0
   (FONTNAME Times)
   (FONTAT R 1.0)
   )
(MAPFONT D 1
   (FONTNAME Times)
   (FONTAT R 0.8)
   )
```

(These definitions go in the preamble, before any character metrics.)

B For each lower-case letter, replace its character metrics with the metrics for the corresponding capital letter, e.g.:

```
(CHARACTER C a      ->  (CHARACTER C a
   (CHARWD R 0.463)        (CHARWD R 0.704)
   (CHARHT R 0.384999)     (CHARHT R 0.6545)
   )                       )
```

C While you’re copying metrics, reduce each of them by your selected scale factor:

```
(CHARACTER C a      ->  (CHARACTER C a
   (CHARWD R 0.463)        (CHARWD R 0.5632)
   (CHARHT R 0.384999)     (CHARHT R 0.5236)
   )                       )
```

(The tedium of this step can be drastically reduced with QuickKeys or some similar keyboard macros.)

D Again for each lower-case letter, add a `MAP` entry that first selects the smaller font, then sets the corresponding capital letter:

```
(CHARACTER C a      ->  (CHARACTER C a
   (CHARWD R 0.463)        (CHARWD R 0.5632)
   (CHARHT R 0.384999)     (CHARHT R 0.5236)
   )                       (MAP
                              (SELECTFONT D 1)
                              (SETCHAR C A)
                              )
                           )
```

That’s all it takes. (Actually, you should also correct the kerning for these letters.) The result is not bad, though certainly not a true small-caps font, because the stroke weights of the small caps won’t match the larger characters.

Limitations

You should keep in mind the following limitations when you define a virtual font:

- 16 heights are allowed per font, which can be a problem when you include illustrations in a virtual font.

- Printer memory limits: When you print using a virtual font, all fonts used in its construction will be downloaded to the printer. Thus, a virtual font composed of characters from many fonts could require a large amount of printer memory. Encapsulated PostScript graphics also use considerable memory: placing many of them on a single page (or in a single VF) as "characters" may also exceed available printer memory.

appendix D

For More Information

This appendix contains information on where to go for more information on TEX and Textures. TEX itself is supported by a worldwide network of users' groups and public archives as well as by a fast-growing library of guidebooks. Blue Sky Research directly supports your Textures installation and its correct operation; we do not offer TEX programming consulting or services.

TEX Users' Group(s)

TEX is supported by an international community of user's groups and academic institutions, and many individual TEX users can help out via electronic mail with TEX-related problems. A special interest Usenet newsgroup for TEX can be found on the Internet, called `comp.text.tex`, with people from all over the world contributing ideas, problems, and solutions involving TEX. There is also a Textures listserv that can be subscribed to by sending email to info@email.esm.psu.edu with the message ”subscribe textures” (no quotes) in the body

The U.S. TEX Users Group publishes a journal, a newsletter, and many other materials; they can also refer you to other users' groups around the globe.

TEX Users Group
1466 NW Front Avenue Suite 3141
Portland, OR 97209
Telephone: (503) 223-9994
Email: `TUG@tug.org`
Web site: www.tug.org

Public-Domain Software

To ensure that his system would be widely used, Professor Knuth placed the TEX program itself in the public domain. (Volume D of Knuth's *Computers and Typesetting* is a painstakingly annotated listing of the entire TEX program.) LATEX and AMS-TEX are also in the public domain, and numerous other public-domain offerings have been created over the years to work with TEX in performing a variety of tasks. The best places to discover what is available are the various TEX archive sites that exist around the world.

Archive Sites There are numerous TEX archive sites at universities and research institutions; here we can only list a few of the largest. For a detailed guide to the TEX archive sites and how to access them, see the on-disk document written by Peter Flynn.

There is a collaborating network of archives known as CTAN (Comprehensive TEX Archive Network): two large archives are `ftp://ftp.cdrom.com` , and `ftp.dante.de` . Also check the Blue Sky web pages, at `www.bluesky.com` for current information on CTAN servers.

NOTE: FTP means *file transfer protocol* and is also the name of a program that allows users to transfer files to and from remote sites that are connected via the Internet. “Anonymous ftp” means that you may connect to a remote site with the user name “anonymous” and a password consisting of your e-mail address, and thus retrieve files from that site.

TEX-Index TeX-Index, by David M. Jones, is a vast and comprehensive index of TEX macros, style files, documentation, and much more. It can be downloaded from any of the CTAN sites (see page 221).

Blue Sky Research

At Blue Sky Research, we offer a variety of educational and support services to our customers. These services include:

- A number of books and manuals, available for separate purchase, that will help you learn TEX and some of the popular formats and macros built on the TEX base (see page 224).

- Technical support for installation problems, font and printer questions and Textures-specific questions, such as the use of the `\special` command with Textures. Available by electronic mail, telephone, fax, or mail.

- Technical Support Web Pages provide up-to-the-minute problem solving tools for Textures users 24 hours a day. There is a wealth of information, including problem-solving decision trees, Frequently Asked Questions, free interim electronic updates, new macro packages, and other useful goodies.

- Installed Electronic Help Pages that provide answers to common questions and solutions to installation and usage problems reported by customers. These pages are installed automatically by the Textures installer.

Blue Sky Technical Support To best help you, we need a description of the symptoms of the problem as well as information on your Textures version and machine configuration. When you contact us for technical support, please provide us with as much of the following information as seems pertinent to your problem:

- Your name and Textures serial number, visible in the About Textures window.

- Your electronic or physical address, telephone or fax number (including the country) so we can respond to your request.

- Your Textures version, including any number or letter extensions; you can get this from Get Info on Textures itself, or from the About Textures window.

- The model of Macintosh you are using and its System file version.

- The version of the Adobe Type Manager (ATM) software you have installed.

- The manufacturer and model of the printer you are using and the amount of RAM it has.

- A full description of the symptoms of the problem, especially whether the problem is present only during screen preview, only during printing, or both.

- The sequence of events prior to the problem.

Referral Services We can refer you to a number of organizations that offer specialized or personalized services when you need to learn to use TeX or upgrade your macro-writing skills, or for complex TeX programming services. Most classes and consultants' services are offered on a fee-for-service basis. Contact us for:

- A list of TeX consultants.

- Telephone numbers and addresses for organizations responsible for writing and maintaining specific macro sets, such as the American Mathematical Society (AMS-TeX) or the American Physical Society (REV-TeX).

- Information on classes offered by TUG or other organizations.

Where to Find Us

Blue Sky Research
317 SW Alder
Portland, Oregon 97204
USA

Phone: 800 622-8398
or 503 222-9571

Fax: 503 222-1643

e-mail:
`sales@bluesky.com`
`help@bluesky.com`
`suggest@bluesky.com`

FTP access:
`ftp.bluesky.com`
World Wide Web Pages:
`www.bluesky.com`

We highly value your letters, faxes, and phone calls. Please let us hear if you have suggestions, complaints, or praise—or when you discover interesting things to do, or interesting ways to do things, with Textures.

Other Vendors

A number of other vendors provide products and services that may be useful to you. Blue Sky Research does not provide TEX implementations for non-Macintosh platforms, but we can refer you to people who do.

Adobe Systems: Creators of the PostScript language, Adobe Type Manager (ATM), and the Illustrator application, and suppliers of fonts for use with PostScript printers.

Adobe Systems, Inc., 1870 Embarcadero Road, Palo Alto, CA 94303   Telephone: (415) 852-0271

A TEX Bibliography

This section lists books that will be helpful (or necessary) if you need information about the following areas. These books, and others, can be ordered directly from Blue Sky Research.

TEX Itself

Paul W. Abraham, *TEX for the Impatient*, Addison-Wesley Publishing Co., 1990.

Michael Doob, *TEX: Starting from 1*, Springer-Verlag, 1993.

Victor Eijkhout, *TEX by Topic*, Addison-Wesley, 1992.

Donald E. Knuth, *The TEXbook*, Addison-Wesley, 1986.

Raymond Seroul & Silvio Levy, *A Beginner's Book of TEX*, Springer-Verlag, 1991.

Wynter Snow, *TEX for the Beginner*, Addison-Wesley, 1992.

LATEX

Leslie Lamport, *LATEX, A Document Preparation System, Second edition*, Addison-Wesley, 1994.

Goosens, Mittelbach, and Samarin, *The LATEX Companion*, Addison-Wesley, 1994.

Helmut Kopka & Patrick Daly, *A Guide to LATEX*, Addison-Wesley, 1993.

AMS-TEX

M.D. Spivak, *The Joy of TEX*, American Mathematical Society/Addison-Wesley, 1986.

PostScript and Encapsulated PostScript

PostScript Language Reference Manual, Second edition, Addison-Wesley, 1990.

Peter Vollenweider, *Encapsulated PostScript, Application Guide for the Macintosh and PC*, Carl Hanser Verlag & Prentice Hall, 1990. (First published in German as *EPS-Handbuch: Encapsulated PostScript*, Carl Hanser Verlag, 1989.)

glossary

Coming to Terms

accent A punctuation mark that appears above a character in a line of type. For example, the TEX **control symbol** `\"` preceding the letter “O” will produce an accent over the letter (Ö) in the typeset version.

Adobe Font Metrics (AFM) file A file shipped with Adobe PostScript fonts which contains **font metrics information**.

Adobe Type Manager (ATM) An Init/Control Panel device that reads PostScript outline (printer) fonts and produces smooth, properly scaled output for your screen or for other non-PostScript output devices. ATM is available from Blue Sky Research or from Adobe Systems, Inc.

AMS-TEX A TEX **macro** package developed by the American Mathematical Society to simplify the input of mathematical material with TEX and to format the output according to preset style specifications. AMS-TEX has been superceded by the AMS-LATEX1.2 package for LATEX.

arrow keys The directional or cursor keys on the Macintosh keyboard that allow you to move the insertion point within a window.

ATM The **Adobe Type Manager**.

backslash The keyboard character “, which is reserved for special use in the TEX language. The backslash is used as an **escape character** in TEX, and it precedes any command that controls the format of your document.

baseline The horizontal line on which a row of characters appears to sit. The space between lines of type (“baselineskip”) is measured from one baseline to the next.

baselineskip The distance between baselines (called **leading** in traditional typography). In TEX, the `\baselineskip` command defines the distance between lines of text in a paragraph.

Bibtex A utility program for producing bibliographies with LaTeX.

bitmap font A Macintosh font stored as the bit images of every character in the font and used for screen display or for printing on dot-matrix printers. (For printing to higher-resolution printers, bitmap fonts have been largely replaced by **outline fonts**.)

box A page-construction unit of TeX. Everything on a page that has been typeset by TeX is made up of boxes; the variable space between boxes is called **glue**. Boxes are rectangular, and have three associated measurements: *width*, *height* (above the **baseline**), and *depth* (below the baseline). For example, in a box composed of a single character, the amount of space that its descender projects below the baseline is that box's depth. The simplest kind of box is a single character, but a box may be anything that TeX treats as a unit.

braces In TeX, braces (`{ }`) are used to specify (1) arguments to commands, and (2) grouping.

break The end of a line, paragraph or page. TeX organizes a sequence of words into individual lines and pages of specified sizes by selecting the optimum places for breaks to occur.

Chooser A desk accessory supplied by Apple and used to select any printer for which you have a printing resource (printer driver) and to designate the port to which your printer is connected.

Clipboard A temporary file created by the Macintosh system software that holds whatever data was most recently cut or copied.

club A single line of type that appears alone on a page preceding the rest of the paragraph of which it is a part. (Also called an *orphan*.)

CM **Computer Modern**.

Computer Modern Designed by Donald Knuth using his METAFONT type design system, Computer Modern is a font family that is TeX's standard font set and is built into Textures. Computer Modern fonts include a comprehensive set of mathematics characters and are good for general text and body copy.

context information Information saved with a Textures document, including the document's **format**, the state of Flash mode (on or off), and several other settings. The next time you open the document, you will again see it in the form that you saved it. **Default settings** can be set for any new Textures file that you create.

control sequence (Also referred to as a formatting command or simply as a TEX command.) A TEX command, always preceded by a **backslash** (or designated "escape character"), that tells TEX what operation you want it to perform on your document. There are two types of control sequences: control symbols and control words.

control symbol A TEX command that consists of a backslash (or designated escape character) followed by a single nonletter. For example, the control symbol `\"` preceding the letter "o" (`\"o`) will produce an umlaut accent over the letter (ö) in the typeset version.

data fork The part of a Macintosh file containing the file's data; in a Textures text file, the data fork contains the document's source text. See also **resource fork**.

default settings **Context information** for your Textures files that you save under the filename **Defaults** in the TeX Formats folder. Textures will use the default preferences you define each time you create a new document.

descender The portion of a character (such as a g or y) that extends below the **baseline** of a row of type.

design size The size at which the characters of a font were designed; this is also the size at which characters in the font will be drawn with maximum quality. Other sizes are **scaled** (magnifying or shrinking the character images). Well-designed fonts, such as the Computer Modern fonts, will be drawn differently at different point sizes, and the letters will often have different relative heights and widths in order to enhance readability. An 18-point CM font, therefore, does not have the same appearance as a 9-point font simply scaled to twice its design size. See, for example, page 16 of *The TEXbook*.

desk accessory A "mini-application" that is accessed via the Apple menu.

dot matrix A regular pattern of dots. This term is often used to refer to printers that, instead of printing formed characters, print an array of dots. It is conventionally used to refer to "dot-matrix printers" such as the Apple ImageWriter, but laser printers and photosetters also print with a series of dots.

download With **PostScript fonts**, "downloadable fonts" are those stored in the computer, as distinguished from fonts such as Times and Helvetica that are permanently resident in the LaserWriter's read-only memory. Before a downloadable font can be used for printing, it must be transmitted, or "downloaded," to the printer. (Textures handles this automatically, except in the case of fonts in included graphics.)

dpi Dots per inch: the resolution of the screen, printer, or other output device.

\dump The primitive TEX command used to create "canned" **format files**. The `\dump` command saves all of the macros that you have defined in a document and takes the place of a `\bye` or `\end` statement.

DVI file A "Device Independent" file which Textures can exchange typeset files with TEX on other computers. DVI files are compact, binary descriptions of typeset pages that conform to a standard representation that can be read by different computers.

Edit window The window where you enter and edit your document's text together with the TEX commands that specify how the document should be typeset. The Edit window is the document's master window, the window that first appears when you open a Textures document.

EdMetrics A utility program included with the Textures package that allows you to edit **font metrics information** and to create font metrics resources.

em A conventional unit of measure that is related to the size of lowercase letters in a particular font of type; hence its size will vary with the font size. It gets its name from the letter "M" which, in early fonts, was usually cast on a square body.

em dash A single-character dash with a width of one "em." In TEX, an em dash is produced by typing three dashes (or hyphens) in a row (---). Em dashes are used for punctuation in sentences—like this.

em space A space having the width of one "**em**" in a particular typeface. An "en" space is usually one-half the width of an "em" space.

en dash A single-character dash with the width of one-half an em dash. En dashes are used to separate numbers in a range, as in the citation, "pages 13–34." In TEX, an en dash is produced by typing two dashes (or hyphens) in a row (--).

encapsulated PostScript file (EPSF) A standard **PostScript** file format for storing a PostScript graphic. EPS files can (and should) include a Macintosh-format version of the graphic for display on the screen.

escape character A keyboard character that is reserved for special use, usually the **backslash** (“) key. The escape character precedes all commands that control the format of your document and is followed by instructions to TeX.

ex-height A unit of vertical space related to the height of lowercase letters in a particular font of type. For fonts in the Latin alphabet, a lowercase x is by definition one “ex” high.

Finder The application that constitutes the Macintosh desktop, used to manage documents and applications and to get information to and from disks. (See the *Macintosh User’s Guide*.)

Flash mode Textures’ interactive typesetting mode, where your document is typeset immediately as you enter the text and formatting commands. Flash mode is indicated by the traffic signal icon that appears at the upper-righthand corner of the Edit window. The alternative to Flash mode is to defer typesetting until you explicitly select the Typeset **command**.

flush A line of type is said to be “flush” when there is no space between it and a reference line. Text that is typeset “flush left,” for example, is aligned with the left margin of a page.

font A complete set of type of one size and typeface, intended for use in a restricted range of output image sizes in a particular reproducing system. TeX deals with sets of up to 256 characters in any one font.

\font The primitive TeX command that defines fonts for TeX so that you can use them in a document.

font mapping The mapping of TeX’s character codes to the character codes used by QuickDraw or PostScript fonts. Also called *character encoding*.

font metrics information Information about character dimensions, kerning, ligatures, and mapping so that TeX can set type for a font. (Font metrics information is the Textures equivalent of a **TeX Font Metrics** [**TFM**] file on other TeX systems.) In Textures, font metrics information is stored in *font metrics resources*.

font metrics suitcase A type of Textures file containing font metrics information (but not the actual fonts themselves).

format A packaged set of macros (or definitions) that define a particular document style, layout, or form of presentation. For example, the LaTeX format is a version of TEX that defines a general set of styles. Textures allows you to package or store such a format in a “canned” format file by using the `\dump` command.

glue In TEX, *glue* refers to units of vertical or horizontal space that are placed between “**boxes**” in page construction. Glue has three attributes: its natural “space,” its ability to “stretch,” and its ability to “shrink.” In a page of justified type, for example, the glue between boxes (words or characters) can stretch or shrink so that the margins come out straight.

grouping TEX will treat as a unit any part of your document that is set off by the control words `\begingroup` and `\endgroup` or enclosed within **braces** (`{ }`). The use of these special characters allows you to change TEX’s normal conventions temporarily inside such a group without affecting the text outside it. Grouping may also be called *block structure*, and definitions that apply only within a group are said to be “local” to that group.

hyphenation The division of words into syllables by a short dash or hyphen. To produce lines of equal length, TEX prefers to hyphenate words rather than stretching interword spaces too much.

INITEX A variant of TEX that is used to install the TEX program on some systems and to create format files. It is built into Textures, however, and no special application is needed.

input file Also called an *included* file: a Textures text file that is read into another file at typesetting time via TEX’s `\input` command.

justification The process by which the space between the words and letters in a line of type is divided evenly to produce a line that is flush with both left and right margins.

kerning A *kern* refers to the space not occupied by a character in a piece of type. (Literally, “kern” means to carve, and refers to trimming the body of lead type to adjust the letter spacing.) “Kerning” is the process of adjusting the spaces between letters to achieve even, consistent letter spacing. TEX automatically kerns pairs of characters specified by the font designer.

LATEX A document preparation system designed by Leslie Lamport as a supplement to TEX, LATEX adds a collection of **macros** to TEX in order to simplify the page-formatting process. LATEX can be loaded into Textures and used as an alternative format to Plain TEX.

leading The space between lines of type (called **baselineskip** in TEX terms). The word refers to the fact that in traditional cast-metal typography, lead was used to set the space between lines of type.

ligature A combination of letters that is treated as a unit. For example, the *f* and the *i* in *find* are not typeset as separate letters; instead, the ligature *fi* is substituted. TEX substitutes a ligature for this combination and others, such as *ff* and *fl*, as specified by the design of each font.

Log window A window that appears during typesetting and displays information and error messages from TEX about the processing of your text. The TEX log also lets you interactively correct errors that TEX encounters during typesetting.

logical page number An internal TEX mechanism allowing an individual page to be identified by a number of different page numbers. This is useful, for example, for front matter which may have a separate numbering scheme from the main body of a document.

macro A stored set of TEX commands or definitions that is substituted for the macro name when that name is invoked in a TEX file. One macro can be defined to expand to any number of functions (or page-formatting descriptions). You can define macros with TEX's `\def` statement.

Macros menu A Textures menu where you can store your own TEX macros and invoke them as menu items. You can define a separate macros menu for each TEX **format**.

magnification Two types of magnification, or enlarged output, are possible in Textures. You can magnify the *screen image* in your Typeset window by using Textures' `View` menu, and you can enlarge or reduce your entire document (for printing) by means of TEX's `\magnification` command. You can also instruct TEX to enlarge or reduce a particular font.

MakeIndex A program that creates indexes with LATEX.

mapping See **font mapping**.

METAFONT A font design system created by Donald Knuth for use with TEX.

metric information (Or, **font metrics**). The dimensions of fonts (such as height and depth of characters) and other numerical information about fonts (such as kerning and ligatures) that TEX needs to set type. This metric information does not include the actual fonts themselves (that is, the actual bit images or outline descriptions).

MPW Shell A Macintosh command line processor that is part of the MPW (Macintosh Programmer's Workshop) software development system. The MPW shell is a programmable processor, modeled after the UNIX shells, and is therefore well suited for processing one or many Textures text files when complex, repetitious tasks are called for.

Option_keys A file in the TeX Inputs folder that defines the Macintosh Option-key characters so that TEX can process them.

outline font A font that describes individual characters in terms of their outlines, represented as a set of lines and curves (as distinguished from a **bitmap font**). Sometimes called a *printer font*. On the Macintosh, an outline font may be either a **PostScript font** or a **TrueType font**.

pathname The full specification of a file name, beginning with a volume or directory name and ending with the name of the given file. A colon (:) separates each element of a Macintosh pathname. For example, `HD:Current_Articles:EPS_Figures:Figure001`.

phototypesetter A very high resolution output device that sets type by a photographic process or on photographic film.

pica A unit of measure in typesetting in English-speaking countries. There are twelve "**points**" in one pica and 6.0225 picas in one inch.

PICT file A standard Macintosh (i.e., **QuickDraw**) pictures file format. A PICT file may contain either a bitmap ("MacPaint-style") or a spline ("MacDraw-style") graphic.

Pictures window A window associated with every Textures document that can include any illustration or graphic material created by Macintosh applications such as MacPaint or MacDraw.

pixel Short for *picture element*: an individual dot on the screen.

PL A **Property List**.

Plain TEX The TEX **format** that is built into Textures, Plain adds about 600 basic control sequences to the 300 TEX "**primitives**." Plain is only one of countless formats that can be designed on top of TEX's primitives. See *The TEXbook*, Appendix B, for a complete definition of the Plain TEX control sequences. Appendix E in *The TEXbook* contains examples of formats that can be added to Plain for special applications.

point A unit of measure in typesetting used in English-speaking countries. A point is a subdivision of a **pica**: 12 points are in a pica, 72.27 points in an inch. Points are generally used to measure type and the distance between baselines.

PostScript A device-independent page description or graphics language developed by Adobe Systems, Inc. The PostScript language is understood by Apple LaserWriter printers, by numerous phototypesetters, and by other output devices.

PostScript font An **outline font** described in the PostScript page description language and capable of arbitrarily high resolution. Times and Helvetica are examples of widely-used PostScript fonts.

PostScript printer A printer such as the Apple LaserWriter that contains a PostScript processor and uses PostScript to describe the image of each page. Non-PostScript printers can print PostScript fonts (though not PostScript graphics) if **ATM** is present.

primitives The set of 300 basic **TEX commands** upon which all other TEX commands are based.

printer driver Software installed in the System Folder that controls a particular type of printer. The current printer driver and printer are selected via the **Chooser** desk accessory. Apple supplies printer drivers for the various Apple printers; the LaserWriter driver may be used with any PostScript printer.

printer font An **outline font**.

Property List (PL) file The "human-readable" form of a **TFM file** containing a font's metric information. A PL file is the basis for a **Virtual Property List (VPL)** file.

quad A unit of relative measure in TEX. A quad is equal to the width of one "**em**" in the current font.

QuickDraw A set of graphics operators, built into the Macintosh system software, that handles all graphic operations on the Macintosh screen (and for QuickDraw printers). **TrueType fonts** are an extension of the QuickDraw system.

QuickDraw printer A printer, such as the Hewlett-Packard DeskWriter, that uses QuickDraw rather than PostScript to describe the image to be printed. The **Adobe Type Manager** (ATM) is necessary for printing PostScript fonts on a QuickDraw printer.

ragged Type that is not justified is “ragged,” which means that letters and spaces are set uniformly as they fall with no extra spacing added. Whatever space remains from a full line width can fall either on the right side (flush left), the left side (flush right), or can be divided equally on both sides of the type (centered).

RAM Random Access Memory. The computer’s (or printer’s) “working memory,” where information is temporarily stored while you are working on it.

rasterization A raster is a regular array of dots or lines. (*Raster* is Latin for *rake*.) Raster image fonts, for example, are not formed but are drawn as a pattern of dots. **Rasterization** is the process of fitting an image to the grid of a particular output device.

resource In the Macintosh system, data packaged in a special format and used by an application. For example, icons, menus, pictures, and fonts are stored as resources.

resource fork (Or, **resource file**.) The part of a Macintosh file that contains **resources**. (The other part of a Macintosh file is the **data fork**.) Once you typeset a Textures document, a description of the typeset document is saved in the resource fork. The file’s resource fork also contains any pictures that are held in the Pictures window as well as the current **context information** for the file.

ROM Permanent read-only memory in the computer or in the printer. In the Macintosh, the ROM contains the built-in system software. In a PostScript LaserWriter, the ROM contains the PostScript interpreter as well as the built-in fonts.

scaling Magnifying or shrinking a particular font in order to use it at different point sizes.

Scrapbook A Macintosh desk accessory that you can use to hold frequently used pictures and text. The Scrapbook is especially useful in transferring pictures from a picture source application into Textures’ Pictures window.

screen font A **bitmap font**, such as Chicago or Geneva, intended for screen display rather than for printing.

serif A class of typefaces in which characters are ornamented by short lines that stem from and are at an angle to the upper and/or lower ends of the vertical elements of the character. For example, the two horizontal bases on which the vertical strokes of the letter 'H' rest are serifs.

Slitex A tool for producing overhead transparencies (slides) with LATEX 2.09. It is now obsolete, and has been replaced by the document class "`slides`."

\special A TEX extension facility by which graphics can be handled as extensions to TEX.

style files In LATEX 2.09, style files define styles for individual document types (articles, letters, etc.) that function as "sub-formats" under LATEX. The recent (2e) version of LATEX replaces style files with document "classes" and "packages"; see *LATEX, A Document Preparation System, Second edition.*

System 6, System 7, System 8 Versions of the Macintosh system software. You can find the version number of your system software with the Finder's Get Info command.

TEX A document formatting system and programming language created by Donald Knuth to produce typeset manuscripts. The TEX language is fully documented in *The TEXbook*. Textures is a complete implementation of standard TEX.

TEX command (Also called a **control sequence**.) A command to TEX, always preceded by a backslash (or designated escape character), that tells TEX what operation you want it to perform on your document.

TEX Font Metrics (TFM) file On other TEX systems, a file that contains **font metrics information**. In Textures, "TFM" information is contained in **font metrics resources**.

TEX Fonts folder A folder in the Textures folder that is used for storing **font metrics suitcase** files.

TEX format See **format**.

TEX Formats folder A folder in the Textures folder that is used for storing any **format** files you will use with Textures.

TEX Inputs folder A folder in the Textures folder that is used for storing **input files** (including pictures files) for your Textures documents.

text Words and characters without reference to the font and typeface in which they appear. For example, any words or characters that appear in Textures' Edit window.

text-only file A file consisting only the characters themselves without any formatting information. (This is usually called an *ASCII file* on other computer systems.) Text-only files are easily transferrable between different computer systems and applications programs. Textures processes text-only files to produce typeset output.

TrueType font A kind of **outline font** supported by System 7 and similar in concept to **PostScript fonts**, but described in Apple's **QuickDraw** graphics language rather than in Adobe's PostScript.

type Printed or typewritten characters.

typeface The *design* of a set of type, as opposed to a **font**, which is the tangible rendering of a typeface. (Though in the realm of digital typography, even a font is not truly tangible.) We have, for example, fonts of Computer Modern faces. *Style* refers to a distinguishing visual characteristic of a typeface; italic, for example, is a style that may be used with a number of typefaces. A set of visually related typefaces with differing styles is called a family of typefaces.

typeset The process of setting text in type; in TeX, the processing of an input text-only file to produce formatted (or "typeset") output.

Typeset command The Textures command (in the `Typeset` menu) that causes TeX to typeset a document and bring the TeX Log window to the front. (In `Flash mode`, typesetting is a continuous process, and you don't need to explicitly select this command.)

Typeset window The Textures window that displays your typeset output.

virtual font A "synthetic" font consisting of characters from various different fonts (or from other sources). From the user standpoint, a virtual font looks like any other font.

Virtual Property List (VPL) file An extension of the **PL** file format that provides the framework for a virtual font.

VPL commands A set of commands written by Donald Knuth and used to write VPL files. VPL commands are documented in the file "Knuth VF Paper."

widow A single line ending a paragraph and appearing at the top of a printed page or column. (Also called an *orphan*.)

wrap An option in the `Edit` menu that causes text you type in the Edit window to be confined to the current width of the window.

WYSIWYG "What you see is what you get:" a buzzword meaning that the computer's screen display looks like the printed output.

index

3

Text
Mark Metzler

Design
Principia Graphica

Illustration
Kimberley Lew
Adobe Illustrator

Typesetting and Imposition
Barry Smith
Textures

Printing
Dynagraphics