VTEX/Free 8.02 Manual

March 12, 2003
M. Vulis
W. Schmidt

This is the documentation for the free releases of the VTEX document compiler.

Currently, there are freeware versions for OS/2 (VTEX/2), Linux (VTeX/Lnx) and Solaris. If this document refers to all of them, it will use the generic term VTEX/Free. If the document refers to something which is also true for the commercial Windows version, it will just use the bare VTEX.

8.02 New or changed features of the program are marked appropriately. Please, notice the new chapter 6

Contents

1 What is VTEX/Free?

VTEX/Free is a partial port of the VTEX/Windows TEX compiler to OS/2, Linux and Solaris. It does not include any shell and/or Visual Tools and there is currently no intention to port those. Even the port of the compiler itself is partial. Out of the five modes of VTEX/Win, only (**PDF**, **PS** (PostScript) and **DVI** are supported, but not **HTML** and **SVG**). The DVI mode is essentially useless, since the main advantages of VTEX's DVI mode under Windows rely on the VTEX DVI drivers which have not been ported. Thus, for all practical purposes, VTEX/Free should be viewed as a PDF and PS mode compiler only.

The PDF and PS modes are very similar in the operation; the implementation code is essentially the same and both the lower-level (`\special`'s) and the high-level (configuration files for `graphicx`, `PSTricks` etc.) are almost identical. This documentation, therefore, equally applies to both modes; differences, where exist, are specifically marked.

VTEX/Free *does* include the full PostScript support (GEX) of the Windows version. This includes both inclusion of (encapsulated) PostScript files and inline PostScript, including support for PSTricks, PSfrag, XYpic and GEXX. GEX can be used in either the PDF or the PS mode; in PDF mode GEX acts as an integrated PS→PDF distiller; in the PS mode, it acts as a PS→PS distiller.

The OS/2 port needs OS/2 Warp 3 or later, and at least 16 MB of physical memory. Below these limits performance may be unacceptably slow. The Linux port is for the x86 platform only; it requires Linux version 2.0 and at least 16 MB of physical memory (without X11) or 32 MB (with X11). VTEX/Solaris is for the SPARC platform only and requires Solaris 2.6 or better.

2 Brief VTEX Documentation

2.1 Invocation and usage

The name of the VTEX executable is `vtex.exe` with OS/2, `vtexsun` on Solaris, and `vtexlnx` on Linux. Thus, the syntax of the command line is:

`vtex` [*options*] `@`*format* *file*[`.tex`]

or on Linux:

`vtexlnx` [*options*] `@`*format* *file*[`.tex`]

Solaris syntax is identical with Linux, except for the name of the binary. In contrast to versions of VTEX/Lnx prior to 7.40, you need no longer specify the full path of the executable.

In case the name of the format is not given, it defaults to `vtex` – as opposed to `plain` with many other TEX implementations.

VTEX options are entered on the command line preceded by a dash (–) character; multiple options should be separated by a space. The options that use alphabetic characters are case-insensitive.

☞ Note, that the distribution comes with two shell scripts named `vlatex` and `vlatexp`, which provide all necessary options to run VTEX in PDF or PostScript mode with the LATEX format. You may want to modify these scripts according to your personal needs, e.g., specify fast static path caching (see section 2.4.2).

VTEX supports the following command-line options:

`-d` Selects the DVI mode (default). DVI files generated by VTEX are compatible with MicroPress' DVI drivers only, unless the `-c` option (see below) is also selected.

`-$p` Selects the PDF mode; see section 3.2 for details.

`-$s` Selects the PostScript mode: see section 3.2 for details.

`-$g` Selects SVG mode. Since there are no reliable SVG viewers available on the Unix or OS/2 platform, SVG mode is not (yez) described in this manual.

`-c` Compatibility mode: Almost all VTEX extensions are disabled. DVI files created in compatibility mode are compatible with most non-MicroPress drivers. *Not useful for PDF and PS modes!*

`-i` IniTEX mode.

`-n#` where # is the initial TEX running mode:

0 batch

1 nonstop
2 scroll
3 errorstop (default)

`-q` Quick run. This switch disables the output. While you need to compile LaTeX documents at least twice (and, often, three times), there is really no need to produce the formatted output on the passes before the last one. Using the `-q` switch will result in approximately 50% time saving on a PDF mode compilation, which would mean 25% overall time saving on a two-pass LaTeX compilation with `-q` used on the first pass, and 33% overall time saving on a 3-pass LaTeX compilation with two dummy runs.

`-2` Enable bi-directional typesetting (TeX-XeT).

`-p` Path caching options, see section 2.4.2

`-s`*dir(s)* appends extra directories(s) to the include path; the directories are searched *after* those specified in the configuration file. This is particularly useful when making a format, because IniTeX does not honor the format-specific include paths.

`-ov` Resolve `.vf` files. In PDF and PS mode, this option is necessary to use virtual fonts. In DVI mode, this option will result in `.vf`-free `.dvi` files.

`-oc` Include complete fonts. This option applies to the PDF and PS modes only and disables font subsetting. In most cases there is no reason to use this switch, since it leads to larger (often, much larger) output. However, if you intend to post-process the documents with Adobe Acrobat (Exchange), you may have to use this switch due to the general buggyness of the Adobe handling of subsetted fonts.

`-j` "Join" fonts. The option applies to PDF and PostScript modes. When a Type 1 font is used both in unmodified and modified (e.g., slanted) form, it will be embedded only once, thus saving space. However, additional code is required for switching between the two variants then. In long documents, this may occur very often, so that they may become actually longer.

`-ox` Enable GeX.

`-ox2` Enable GeX with PostScript Level II support.

`-ob2` Make GeX recognize all 35 "base" fonts. (Default is to support the "Base 13" only.)

`-e` Security settings; see section 3.13.

7.59 `-of=`*file* Specify the FontMap file, see section 2.3.1.

`-ofr=`*file* Use compiled FontMap; see section 2.3.7.

`-ofw=`*file* Compile FontMap; see section 2.3.7.

`-=`*file* Specify alternative main configuration file, see the below section 2.2.

`-83` Disable use of "long" file names with VTeX/2, e.g., for use with a FAT file system.

See section 3.2 for further options related with PDF or PS generation only.

2.2 The main configuration file

Search paths and some general settings are determined by a "general" configuration file, whose name is `vtex.ini`, on both Unix and OS/2. With OS/2, the file resides in the same directory as the VTEX executable. With Unix, it is searched first in the directory `$HOME`, then in `/etc`. A different configuration file can be specified through the option "`-=`" on the command line (see above).

The main configuration file is a typical Windows-style INI file, which is divided into several sections, each defining its variables. Several sections are not applicable to the VTEX/Free distributions and have been omitted from this description.

2.2.1 Paths

The section `[Directories]` defines program paths. The following paths are important:

`PGMDIR` : VTEX binaries, default `vtex/bin/`
`INCDIR` : generic TEX include path, default `vtex/src/`
`FMTDIR` : Format directories, default `vtex/fmt/`
`TMPDIR` : Temporary directory, default `vtex/tmp/`
`GRFDIR` : Graphics file include directories, default `vtex/src/`
`TFMDIR` : `.tfm` include directory, default `vtex/tfm/`
`VFSDIR` : `.vf` include directory, default `vtex/vfs/`
`ENCDIR` : `.enc` include directory, default `vtex/enc/`
`BIBDIR` : BibTeX database files (`.bib`)
`BSTDIR` : BibTeX style files (`.bst`)
`MSTDIR` : MakeIndex style files

Multiple directories should be separated with semicolons, and the trailing path separator must be included as well. Automatic subdirectory searches can be turned on on any of these directories by appending a + to the directory name.

The second important section is `[FINCLUDE]` The `.tex` include path in VTEX is format-dependant. Assuming that you compile with the "`latex`" format, VTEX will check the `FINCLUDE` section to see if a particular include path for this format is listed; if it is, it will be used rather than the generic `INCDIR` setting. In IniTEX mode, however, only the generic include path is used; the name of the format to be generated is *not* honored.

Note that the `FINCLUDE` directory match works by prefix comparison. This means that if your format is, for instance, `latexhv` (HV-Math based LATEX 2ε), VTEX will first check `[FINCLUDE]` for `LATEXHV=`, if this fails, it will try `LATEXH=`, followed by `LATEX=` etc. With the default settings, the match will be achieved at `LATEX=`. The purpose of this logic is to allow compact specifications for a set of related formats.

The include path defined in the `FINCLUDE` section may reference the generic include path (`INCDIR`) with the * symbol. By default, we define

```
VTEX=*
PLAIN=*
LATEX=/vtex/l2e/;*
```

which means that Plain-based or VTeX-based documents search only `/vtex/src`, while LaTeX documents first search `/vtex/l2e`, and then `/vtex/src`.

Automatic subdirectory searches can be turned on on any of these directories by appending a + to the directory name

2.2.2 TeX memory

In the section `[HUGETEX]`, TeX's initial `hyph_size`, `trie_size` and `trie_op_size` memory can be specified (in bytes). The hyphenation memory sizes can grow dynamically. If needed, `mem_size` can also be tuned; note that you have to specify it in units of 64k memory words, e.g., `mem_size=16`. The default value is 8; changing this would require a format rebuild.

2.2.3 Return codes and PS interpreter memory

The `[COMPILER]` section deals with further internals:

`retcodes=n1,n2,n3,n4` allows to modify the default return values. The four numbers correspond to the return code for clean compilation, warnings, errors and fatal errors (cf. TeX's history). The assumed default is `retcodes=0,1,2,3`. The configuration file supplied with VTeX/Free sets `retcodes=0,0,1,1`, which is suitable for use under *Make*.

7.49 `VMSIZE` allows to specify the memory size of the PostScript interpreter. It defaults to 5 MB, if no setting exists in `vtex.ini`. This is is usually sufficient, but if GeX crashes on "vmerror", you can increase the value, e.g., `VMSIZE=6000000`.

2.3 Configuring the font usage

2.3.1 The FontMap (FM) file

Font usage in the PDF and PostScript modes is primarily controlled by a configuration file, whose name is usually given on the command line using the option `-of`; VTeX will search for this file along the TeX include path.

The FM file is made up from a number of "sections". Each section is started by a keyword, followed by the contents of the section included in braces. The particular sections are:

`LOCAL-OS2`, `LOCAL-LNX` These sections define local variables, which can subsequently be referred to in the font mapping ("aliasing") files; `LOCAL-OS2` is evaluated with OS/2 only, whereas `LOCAL-LNX` is for both Linux and Solaris. For instance, the FM files supplied with VTeX/Free define in the `LOCAL-LNX` section:

```
TEXMF = "/usr/local/vtex/texmf/"
URW = "/usr/share/ghostscript/fonts/"
```

The LOCAL-... sections allow to use the same FM file on different platforms in dual-boot environments.

PSRENAME Here the name of the file is to be given, which contains the list of base font substitutions; see section 2.3.3.

TYPE1 This section lists the font aliasing files for Type 1 fonts.

TRUETYPE This section lists the alising files for TrueType fonts.

OPENTYPE This section lists the aliasing files for OpenType fonts.

VF3 This section lists optional mapping files for virtual fonts; see the below section 2.3.6.

The files listed here must reside in the TEX include path.

When no FM file is given on the command line, a number of default settings come into effect:

LOCAL... No local variables are defined.

PSRENAME The file name defaults to `fontname.fix`

TYPE1 Only one single aliasing file named `aliasing.pst` (Win, OS/2) or `type1.rc` (Unix) will be used.

TRUETYPE Only one single alising file named `aliasing.tt` will be used.

OPENTYPE Only one single aliasing file named `aliasing.ot` will be used.

VF3 Only one single mapping file named `vfwrap.fm` will be used.

2.3.2 The "aliasing" files

These files establish the relation between the font names used by TeX (i.e., the TFM's) and the physical font files (Type 1, TrueType or OpenType). On other TEX systems they would be called "font map files". The files are ASCII and can be manually edited if needed. VTEX will search for these file along the TEX include path.

An aliasing file is made up from two sections, separated by an empty line.

The first section lists the directories that include the font files referenced in the second section. Each line is in the format

```
%<N> = <directory>
```

where <N> is a unique (non-repeating) integer number and <directory> is a full directory path, ending with a slash. For example,

```
%1 = $PFM$
%2 = $TEXMF$fonts/type1/micropress/cm/
%3 = $TEXMF$fonts/type1/micropress/ams/
```

defines three directories that can be referenced later through their number. Note, that the slash is used as a directory separator with OS/2, too! This is not a requirement, but using slash allows to maintain the same `.ali` files on Unix and OS/2. `$TEXMF$` refers to the variable `TEXMF`, which must have been defined in the FM file – see above. The variable name `VTEX` is predefined and points to the VTEX root directory. (This is relevant for VTEX/Win. only.)

The second section of the file contains the list of fonts. Each line has the format

```
@<Font-name> = <File-name> %<access><File-dir>,<pfm-dir>[options]
```

for Type 1 aliasing files and

```
@<Font-name> = <File-name> %<File-dir>[options]
```

for TrueType and OpenType `.ali` files,

where

`FontName` The PostScript FontName of the font e.g., `Times-Roman`. This name *must* match the name given in the font file exactly.

`File-name` The `.tfm` file name for the font as referenced by TEX. It must be the same as the name of the related font file, unless the latter is explicitely specified.

`access` Usually, nothing. Fonts that are built-in in the PDF or PostScript interpreter ("base fonts") should have a `!` mark to prevent their (unnecessary) downloading into the document. Fonts that are to be embedded in full must be given a ~ mark. Full embedding is an escape provision in case you encounter a font which VTEX cannot subset. (Shall this happen, please let us know.)

`File-dir` The number of the directory that contains the file for this font.

☞ VTEX cannot search for Type 1 fonts recursively in a given directory (sub)tree! The default extension of the file is `.pfb` for Type 1, `.ttf` for TrueType and `.otf` for OpenType fonts.

`pfm-dir` The number of the directory that contains the `.pfm` file for this font. This applies to Type 1 `.ali` files only; the presence of the `.pfm` file is not required, and this entry is in no way used by the VTEX compiler.

Additional options that can be specified include

`b=file-name` The name of the actual font file (basename), in case it differs from the TFM name; the extension (`.pfb`/`.ttf`/`.otf`) may be omitted.

`r=map-name` Reencoding map file; the extension (`.enc`) may be omitted.

`g` GEX-loading: full processing of the font (Type 1 only) in GEX. This option allows to use Type 1 fonts that are not valid ATM fonts.

`m=...` MM flags (Type 1 only): See the document `MMsupp.pdf`, which is supplied with VTEX for Windows, for more details.

Examples:

```
@cmr10 = cmr10 %2,1
@Symbol = psyr %!2,1,b="s0500001"
```

Note that the "Symbol" font is built-in and so marked.

☞ In VTEX/Free, the `.pfm` directories are never actually accessed, but because the syntax for this file is identical to the Windows version, you need to define at least a dummy directory.

2.3.3 Base font substitition

The font substitution table allows to use non-Adobe "Base 35" font files. For copyright reasons, these fonts usually carry incorrect /FontName's; this may cause the problems as follows:

Assume you have an `.eps` file which uses Palatino, but does not embed it. Loading URW's Palatino clone will not help, since its /FontName is hard-wired as `/URWPalladioL-Roma`, while the `.eps` program would look for `/Palatino-Roman`.

GEX overcomes the problem as follows: When the font is first loaded, its `/FontName` is replaced. The replacement table is given in the `fontname.fix` file; each line of this file lists two names: the name to be replaced first, and the replacement second. For example, the Palatino substitution is defined as

```
/URWPalladioL-Roma      /Palatino-Roman
```

☞ Note that the Times, Helvetica and Courier fonts are special: they must *not* be listed here, and one may "silently" load *only* URW's or MicroPress' clones for them.

2.3.4 Re-encoding

Type 1 and OpenType fonts typically contain more characters than accessible directly. These additional characters, "hidden" inside the font implementation, include ligatures (usually, "fi" and "fl" only), accented characters, and others. The visible characters are only a subset of what can be made available. To make things worse, the *default* layout of third-party fonts (e.g, `StandardEncoding` in the case of Type 1 fonts) makes only a small and almost useless subset of the font immediately available. The layout of the CM fonts which are supplied with VTEX/Free is, however, already suitable for TEX and no re-encoding needs to be applied to these fonts.

VTEX allows you to configure Type 1 and OpenType fonts to follow a desired character layout with full ligature support. VTEX comes with several encodings, including `TeXBase1Encoding` (file `8r.enc`), which is used by the virtual fonts of the PSNFSS system. Additional encodings can be easily made by the user. See the next section for the details on the format of these files.

Since there may be more than one TEX font based on a single physical font and multiple encodings, they have to be distinguished by different font names and TEX metric `.tfm` files. Note that the font names can be chosen arbitrarily in this case. With fonts to be reencoded, the entries in the aliasing file contain an additional option to support this:

```
r="<encoding file name>"
```

The extension `.enc` of the encoding file may be omitted

☞ For each reencoded font there must also be an entry which references the "raw" (not reencoded) font under its actual name. Otherwise the font cannot be used in GEX, i.e. within PSTricks, PSfrag, in-line PostScript code etc.

Example: The following lines within a font map file declare two instances of the Charter-Roman font, “raw” and reencoded:

```
@CharterBT-Roman    = bchr8a %2,1
@CharterBT-Roman-8r = bchr8r %2,1,r="8r",b="bchr8a"
```

2.3.5 Encoding files

The available encodings are stored in files with extension `.enc` within the ENCDIR subdirectory. Each file should declare a PostScript array of exactly 256 symbol names:

```
/MyEncoding [
        /.notdef /.notdef /.notdef /.notdef
        ....
        /A /B /C /D /E /F
        ....
        /a /b /c /d /e /f
        ....
        ] def
```

Comment lines are allowed and should start with the % character. These names define the layout of the reencoded font. Note that if you specify a symbol name that actually does not occur in the base font, the entry will be blank (`/.notdef`).

The number of entries in the encoding vector must not exceed 256.

2.3.6 Font mapping files for virtual fonts

These files may list virtual fonts, which are to be converted to Type 3 wrapper fonts, instead of being fully resolved. Each line has the format

```
@<virtual font> % x="<bounding box>",e="<encoding>"
```

where:

`bounding box`: Desired bounding box for the entire font. Optional; if not present, `0 0 1000 1000` will be used. Generally, the default is good enough, but smaller values will produce more efficient output.

`encoding`: Encoding file to describe the encoding of the virtual font. Optional; if not present, glyph names like /X00, /X01 etc. will be used.

The main purpose of this feature is to fully support searching and copying from the Acrobat window with fonts, that are made up from composite glyphs, such as the virtual AE fonts. For instance, the record for AE Roman looks like this:

```
@aer10  % x="-500 -200 500 800",e="ec"
```

Notice that searching and copying *requires* the encoding to be specified.

A minor disadvantage of “VF wrapping” is, that Acrobat Reader 3 is somewhat slow in displaying the resulting PDF files. With respect to the PDF

output size, use of Type 3 fonts for VF's will typically increase the size of smaller documents, but decrease the size of the large ones.

2.3.7 Using precompiled FontMap files

8.02 Beginning with version 8.0, VTEX also supports FontMap precompiling. A compiled FontMap (`.cfm`) contains a binary dump of an (`.fm`) file as well as some additional information extracted from the fonts themselves; use of a compiled FontMap improves the performance of the VTEX compiler.

To compile a font map (for example, `name.fm`), use the

```
vtex -ofw=name
```

command line. This compiles `name.fm` into `mapname.cfm`, which resides in the formats dicetory.

To make VTEX load the information from a compiled font map, use the `-ofr=name` switch:

```
vtex -ofr=name ...  myfile.tex
```

☞ The speed improvement offered by compiled font maps is not very large; use of compiled font maps is recommended mostly in batch production environments.

2.4 File I/O

2.4.1 Reading files

The following directories are searched for files to be read (font metrics, packages, `.tex` files to be included etc.), in decreasing order of precedence, unless an absolute path is specified:

1. The working directory,
2. the directory of the main document, which may be different from VTEX's working directory,
3. the include paths, as configured in `vtex.ini`.

2.4.2 Path caching

While the CPU speed of the modern computers makes the document compilation fast, the file search performed by the operating system could be very slow. As a result, TEX may spend a considerable amount of time searching for files to be read. This bottleneck can be eliminated with *path caching*.

Path caching acts on macros and include files, which are read by TEX, as well as on the TEX font metrics files (`.tfm`) and the virtual fonts (`.vf`).

The path caching in VTEX comes in two models:

▷ Dynamic path caching: at the beginning of each compilation, VTEX builds a full list of the contents of the include directories in memory; when a

file needs to be opened, VTEX uses the memory table to get directly to it, without the need for additional searches.

▷ Static path caching: the cache is built once and stored as a file; on each compilation, the cache is read and is used to locate files. The path cache files have extention `.ffs` and are stored in the format directory of VTEX. Since different TEX formats may have different search paths, you need a separate file for each format you use. For example, `latex.ffs` is the name of the path cache file for LATEX.

Static path cache is more efficient than dynamic, since it is faster to load a cache from disk than to search the entire include paths (even once). However, static path cache may need to be rebuilt when you modify the include paths or add more files to the include directories. Otherwise, searching for files, which have been added or moved, will be unnecessarily slow, or TEX may not find the files at all, depending on the cache settings (see below).

☞ Since the directory of the document is always searched *before* the path cache, a file in this directory is guaranteed to have precedence over a duplicate one, whose location has been memorized in the path cache. As a result, a static cache is not affected, when include files or special macro packages are stored together with a document in one directory.

The command-line options that govern cache operations are:

`-pu` Use dynamic cache.
`-pw` Write static cache file.
`-pr` Read static cache file.

Example:

`vtex` *further options* `@latex sample2e`

compiles the standard LATEX sample file without path cache. This is the slowest.

`vtex -pu` *further options* `@latex sample2e`

compiles the same file using dynamic cache (faster).

`vtex -pw @latex`

writes out the static path cache for the LATEX format. Note that this operation does not compile any document; it just writes the cache and exits.

`vtex -pr` *further options* `@latex sample2e`

compiles the file using the static cache (fastest).

There are two modifiers allowed after the path cache options:

`t` produces a console trace. You can use it to detect duplicate file names and see which of the duplicates will actually be used by VTEX. This modifier can be used with `-pw` and `-pu` (`-pwt`, `-put`).

`0` specifies cache-only search. Usually, if the required file is not found in the cache, VTEX will attempt to find it on disk anyway. With this modifier, VTEX will not spend time looking for the files on disk. This modifier is used with both `-pu` and `-pr`. The additional time savings

from the use of this switch come mostly from non-searching for `.vf` files (which applies only if you specified the `-ov` option).

☞ Using `-pu0` is recommended under normal circumstances. This is the setting in the supplied shell scripts.

☞ `-pr0` saves additional time over `-pr`, but may cause unexpected errors if the static cache file is out of date. For example, if you copied a new style into a directory which is on the include path, with `-pr0` it would not be seen by VTEX until you rebuild the cache. Use this switch only, if you are aware of the consequences!

2.4.3 Writing files

In case there is no absolute path specified, files written by VTEX, such as the auxiliary files for LATEX, are created in the directory of the main document, which may be different from the working directory. The final PDF or PostScript files are, however, written to the working directory.

☞ In PostScript mode, VTEX uses to create a temporary file with the name of the main document and the extension `.~~~` in the working directory, so this extension should not be used for other purposes.

3 PostScript and PDF generation

3.1 General

The VTEX document compiler supports generating PDF and PostScript files in addition to the traditional DVI files.

This has been designed to be totally transparent to use. No changes are required to your document source. All additional features, such as the hyperlinks and outline entries supported by the Portable Document Format, are supported via `\special` commands, rather than syntax changes.

The PDF and PS backends also support *most* of the VTEX extensions. This chapter explains most of the PDF/PS mode specifics. Omitted are the more interesting parts, which are explained in separate manuals:

- ▷ GEX, the integrated PostScript→PDF compiler, see gex.pdf .
- ▷ Form and JavaScript support, see forms.pdf .

Most features shown in the latter document require Acrobat 4 or better!

☞ The PDF-specific features explained later in this chapter, such as hyperlinks or the PDF outline, are disabled in PostScript mode, unless VTEX has been called with the `k` subswitch – see below. The related `\specials` or use of the `hyperref` macro package have no effect. *With* the `k` option, pdfmark commands are generated. This is useful, if you intend to post-process the document with a PS→PDF distiller. The shell scripts for running VTEX in PostScript mode do *not* enable pdfmark generation by default.

3.2 Command-line options

To create a `.pdf` file, you should supply the `-$p` option to the VTEX document compiler. For example (OS/2)

vtex [*options*] -$p[(*subswitches*] @*format file*[.tex]

or (Linux)

vtexlnx [*options*] -\$p[{*subswitches*] @*format file*[.tex]

The counterpart for the PS mode is the `-$s` option witn the same syntax.

Under most Unix shells you will need to precede the `$` with the `\` escape, as shown above. Alternatively, you can use `@` instead of `\$`. Furthermore, some Linux shells are unhappy about `(` as well; you may use `{` instead, as indicated.

The *subswitches* part includes the options (switches) that are specific for the PS and PDF mode. Note that this part must not contain embedded spaces;

all options are separated by commas. The mode-specific switches are

c `c=<number>` specifies the compression level (0 through 9). This option affects only Flate-compressed parts of the document. Higher number indicates stronger compression (meaning: smaller `.pdf` file and slower compilation).

d `d` specifies to end lines in `.pdf` files with the DOS [CR][LF] sequence, rather than the default [CR].

! Do not load the "base" Type 1 fonts into GEX unless they are actually required, thus speeding up GEX.

f `f=<output spec>` specifies how fonts should be embedded into the document.

g `g=<output spec>` specifies how graphic images should be embedded into the document.

h `h=<dimen>` specifies the document MediaBox or BoundingBox height.

k enables generation of pdfmark commands (PostScript mode only).

o Space optimization (PDF mode only); see section 3.15.

t `t=<output spec>` specifies how text should be embedded into the document.

v makes VTEX embed a font one time only, even if it is used with multiple encodings.

w `w=<dimen>` specifies the document MediaBox or BoundingBox width.

z forces emission of an empty AcroForm record (PDF mode only). As a result, the Acrobat Reader 4 on Linux can reload the PDF file without having to unload it first. (On OS/2 or Windows the z switch is useless.) The switch is intended to be used *only* during the development of your document, not for the final version. Final versions of PDF files intended to be distributed should be recompiled without the z switch. Files that are compiled with the z switch and also use JavaScript may exhibit problems with some versions of Acrobat 5 on Windows.

_ `_=<char>` replaces underscores in internally generated font names with the indicated character, for compatibiliy with Acrobat Reader 3; this is currently relevant in conjunction with MM fonts only.

The `<output spec>` may include these letters:

a ASCII85 output

h Hex output

b Binary output (default)

c CFF conversion of Type 1 fonts (with `f=` key only), see section 3.4.

n No compression of the output (default)

f Flate compression of the output

In the case of the conflicting options (for example, `n` and `f`), the last specification is taken. Thus, `f=nb` is the same `f=b`. The default options do not need to be specified.

The following specifications are particularly useful:

- `f=a,g=a,t=b` : when used in PDF mode, this will create an ASCII-editable `.pdf` file. Note that `t=b` is not a misprint: the text portion of the generated `.pdf` file always contains only printable characters and does not need to be further encoded. Note further, that here "ASCII-editable" `.pdf` files means that such files *usually* can be loaded into a text editor, changed in a minor way, saved, and then loaded into the Acrobat Reader, which should be able to repair them. *usually* does not mean *always*, of course. Naturally, to make manual corrections to the `.pdf` format you should know the format.
- `f=cf,g=f,t=f`: this will create the smallest `.pdf` file.

Use of Flate compression in the PS mode is supported, but should be restricted to the cases when the PS output is to be processed by a Level III processor. These include GhostScript, Acrobat Distiller 4 and above, but not the majority of the printers.

3.3 Font usage in PDF and PostScript documents

VTEX-built PDF and PostScript files currently support these font formats:

Type 1 This is the recommended format.

IF4 VTEX native fonts; currently not supplied with VTEX/Free.

TrueType In general, we strongly recommend *not using* TrueType fonts. Some reasons for this are:

- Type 1 fonts almost always deliver higher quality.
- TrueType fonts cannot be used in/with GEX.
- Currently, TrueType fonts cannot be reencoded.
- TrueType fonts often cause problems with cross-platform usage.

OpenType For the time being, use of OpenType fonts with VTEX is very much theory, because there are no tools to create the required TEX metrics for these fonts. Furthermore, they can be used with PDF mode only, and cannot be loaded into GEX. Future improvements are to be expected in this area.

PDF mode, but not the PS mode, also supports CID fonts (used in CJK).

☞ VTEX/Free does *not* support the METAFONT format.

To avoid generating bad output files, VTEX checks for the presence of an acceptable font file whenever the font is referenced in the document. Further, before starting the compilation of a document, VTEX ensures that all fonts referenced in the format file correspond to valid (i.e. in a supported file type)

fonts. If a valid font is not available, VTEX will generate an error and substitute the `nullfont`. Note that this behavior may produce documents that are formatted differently as `.dvi` and as `.pdf`.

At the time of the font availability check VTEX also decides which format and font file to use. The check is done as follows:

- ▷ If the font is listed in one of the the font map files, it will be used. Otherwise,
- ▷ if the font file with extension `.if4` is present in the `VTEX\IF4` subdirectory, it will be used. (In practice, this is irrelevant for VTEX/Free.)

3.4 CFF conversion

One of the major enhancements in version 7.3 is the ability of the VTEX compiler to convert Type 1 fonts to the new CFF format. The Compact Font Format is much more efficient than Type 1; it cuts down the size of the font files by a factor of 2 or more. Since in typical TEX documents the fonts contribute most to the size of the `.pdf` output, such documents, when recompiled with CFF compression enabled, often would become much smaller.

The operation of the CFF converter is transparent to the end-user; the CFF converter is enabled with the `-$p(f=c` command-line key, or, for the most possible compression, with `-$p(f=cf`: this will perform Flate compression on top of the CFF conversion. and is the default setting in the shell scripts `vlatex` and `vplain`.

CFF conversion can be performed in either PDF or PS modes; however, while in the PDF mode, the PDF readers (both Acrobat Reader and GhostScript) understand embedded CFF fonts, in the PS mode, GhostScript is the only application which is able to process `.ps` files with embedded CFF fonts. Adobe applications are mostly unable, even if it was Adobe that invented the CFF format and published the specs for CFF embedding into the `.ps` files. CFF embedding in the PS mode generally should be avoided; however, the `-$s(f=c` switch is available for special-purpose processing.

3.5 MediaBox and BoundingBox

The default MediaBox in the PDF file or the BoundingBox in a PostScript file generated by VTEX will be based on the settings of `\hsize` and `\vsize`. You can override this by using the w and h command line options explained above (section 3.2) or from within the document through the non-standard `dimen`'s `\mediawidth` and `\mediaheight`. This has the advantage that the settings are stored with the document.

With LATEX, the media size is automatically determined from the dimensions of the page as specified in document, when `\begin{document}` gets executed:

7.53 Normally, VTEX sets the media size to `\paperwidth` × `\paperheight`. However, if the length `\stockwidth` has been defined, VTEX sets the media size to

\stockwidth × \stockheight instead. Thus, the physical media size (e.g., the untrimmed sheet of paper in the printer) and the "logical" page size (after trimming) can be different. In this case, \paperwidth and \paperheight are to reflect the logical page size (see the documented code of the LaTeX standard classes), whereas \stockwidth and stockheight are expected to indicate the physical media size (this was introduced with the memoir class).

Automatic setting of the media box requires, that the LaTeX format file has been generated from the driver file latex.fid, as suggested in the LaTeX Local Guide for VTeX/Free.

Document classes are not required to define \paperwidth and \paperheight, so a fallback media size should be specified through the command line. (\hsize and \vsize are not correct with LaTeX.) Particularly, the vlatex and vlatexp shell scripts specify the A4 format for this purpose.

To summarize this, the MediaBox or BoundingBox is determined, in descending order of precedence, by

1. \mediawidth and \mediaheight,
2. \stockwidth and \stockheight (with LaTeX only),
3. \paperwidth and \paperheight (with LaTeX only),
4. the command line options w and h,
5. \hsize and \vsize.

3.6 Links

One of the advantages of .pdf files is the ability to produce hyperlinks. On the low level, this is accomplished by VTeX \special commands:

!aref... Start a hyperlink

!endaref End a hyperlink. This \special has no arguments.

!aname Define a target.

Both \special{!aref... and \special{!aname... have additional arguments. The most important one is the label specification which should be the first and has one of three formats:

*<Number> This denotes a local (non-exportable) label.

Name This denotes a global (exportable) label. Name generally should contain only alphanumeric characters. Name resolution is case-sensitive.

!<Number> This denotes a page number label.

Note that global labels are accessible from other .pdf documents while local labels are not; on the other hand, local labels result in somewhat smaller document files. Note also VTeX always generates a label PageNNN on each document page, which can be used for external references.

Example of a simple hyperlink follows:

```
See \special{!aref Gnues}Gnues\special{!endaref}
for more details.
...
```

```
\special{!aname Gnues}\section{Gnues}
```

To produce a link to another `.pdf` file, use the global (or page) label format and precede the name with <f*filename*> specification. For example,

```
See documentation on
\special{!aref <f=pdfspec.pdf>!8}pdf\special{!endaref}
        %% Refer file pdfspec.pdf, Page 8.
file format for more details.
```

Note that if you do not provide the target file for an external link given in this form you will a warning

```
Invalid file specification object.
```

from the Acrobat Reader. This error does not mean a structural problem in the `.pdf` file, but merely an absense of the target file.

The appearance of the link itself can be controlled in two ways:

- ▷ Via TEX itself, you can, for example, underline the link, or color it, or produce a colorbox over the link.
- ▷ Via `.pdf` syntax you can specify the attributes of the link. The attribute specification can appear in the `!aref` command after a semicolon and `a=`; everything that follows `a=` through the end of the special is passed over to Acrobat as the link attribute. For example,

  ```
  \special{!aref Name;a=</Border [0 0 0]>}
  ```

  will produce a link without the default border. Specifying

  ```
  \special{c"FF0000}\special{!aref Name;a=</Border [0 0 0]>}
  .....
  \special{!endaref}\special{c"000000}
  ```

  will produce red-colored links without border. For more details on link attributes, see the PDF documentation, as published by Adobe.

The `\special{!aref}` and `\special{!endaref}` commands should come in pairs: for each `\special{!aref}` there should be a matching `\special{!endaref}`. Furthermore, a matching pair should be given on *the same TEX nesting level*. For example,

```
...
\smallskip
\special{!aref Label}See section 5
\special{!endaref}
for more details.
...
```

is incorrect and will cause a backend error, since `\special{!aref}` appears in the vertical mode, while `\special{!endaref}` appears in the horizontal. One way to avoid (or minimize) such nesting errors is to always place `\leavevmode` before a `\special{!aref}`; the other is to place everything (`\special{!aref}`, `\special{!endaref}` and the text between them) into an `\hbox`.

In the PS mode, hyperlinks (as well as the outline entries, described in the next section, are emitted as `pdfmarks`. `pdfmarks` are ignored by most of PS processors but processed by PS→PDF converters (distiller, GhostScript in ps2pdf mode). Emission of `pdfmarks` in VTEX's PS mode makes sense only if you intend to post-process the `.ps` output and then feed it to a PS→PDF converter.

☞ With LATEX we recommend to generate hyperlinks through the `hyperref` package, rather than directly using the above low-level commands.

3.7 Outline

You can supplement the `.pdf` file with an outline. The low-level interface for this is the VTEX `\special{!outline}` command. The general syntax of this command is

```
\special{!outline Label;i=Id,p=Parent,s=Status,t=Text,c=rrggbb,f=#}
```

where

- Label is as described in the previous section (local, global, or page). For local or global tags you should have a defining `\special{!aname ...}`.
- The Id (`i=`) is a unique integer number identifying this outline record. You must not have two outline entries with the same id.
- The Parent (`p=`) is the id number of the parent entry, or 0 for the topmost entry.
- The Status (`s=`) is the letter 'o' (open) or 'c' (closed) identifying the initial state of the outline at this node. It is used only for the non-leaf nodes of the outline (i.e. the nodes that have subchildren). Due to bugs in many version of Acrobat, we strongly recommend keeping all entries closed.
- The Text (`t=`) is the text for the outline. Since this text is not formatted by TEX, you should use only "normal" characters within it. For example, putting `\TeX` within the outline entry text is a very bad idea.
- The color of an outline entry can be specified with the `c=rrggbb` key. For example, to specify a red entry, use `c=FF0000`. This feature is of limited use at this time, since versions of Acrobat before 5, as well as all versions of GhostScript ignore color specifications.
- The font style of an outline entry can be specified with the `f=#` key. The `#` value can be 0 (normal, default), 1 (bold), 2 (italic), or 3 (bold italic). For example, to specify a bold italic entry, use `f=3`. This feature is of limited use at this time, since versions of Acrobat before 5, as well as all versions of GhostScript ignore style specifications.

VTEX version 6.12 and later processes the `\char` commands within the outline title specifications. You can use this to place accented letters into the outline entries. An example LATEX file, `otlchrs.tex`, is supplied in the `texmf/doc/vtex/examples` subdirectory.

The `\char` command should be followed by a decimal, hex (with a leading "), or an octal number (with a leading '). Use of decimal is discouraged, since it may lead to errors due to varying size of a decimal number. Hex and octal are safe since they always require a fix number of digits (2 and 3).

☞ If you are a LaTeX user, please consider using a predefined high-level style such as the `hyperref` package by S. Rahtz, rather than programming the low-level commands.

3.8 Page transitions

The information in this section applies to the PDF mode only.

Starting with version 6.4, VTEX supports Acrobat pdf transition effects, which can be used to spice up a pdf-coded presentation.

Transition effects are coded using the `\special{!trans ...}`. It takes the form:

`\special{!trans` <CODE>[,<TIME>]+`}`

The <TIME> part is optional; if present, it indicated the transition time in milliseconds. The possible transition codes are:

- `W0`, Wipe at 0 degree angle.
- `W90`, Wipe at 90 degree angle.
- `W180`, Wipe at 180 degree angle.
- `W270`, Wipe at 270 degree angle.
- `D`, Dissolve.
- `BH`, Blinds, horizontal.
- `BV`, Blinds, vertical.
- `G0`, Glitter at 0 degree angle.
- `G270`, Glitter at 270 degree angle.
- `G315`, Glitter at 315 degree angle.
- `XI`, Box, In.
- `XO`, Box, Out.
- `SHI`, Split, Horizontal, In.
- `SVI`, Split, Vertical, In.
- `SHO`, Split, Horizontal, Out.
- `SVO`, Split, Vertical, Out.

These are the only transition values supported by the Acrobat. An example Plain TEX file, `trans.tex`, is supplied in the `texmf/doc/vtex/examples` subdirectory. This file shows all available transitions.

☞ With LaTeX, page transitions should be specified using the generic `hyperref` package instead of the VTEX-specific `\special{!trans ...}` command.

3.9 Thumbs

The information in this section applies to the PDF mode only.

Starting with version 6.5, VTEX supports page thumbs generation. Page thumbs are shown by the Adobe Acrobat Reader when the "Thumbnails" sub-Window is open.

VTEX supports two `\special{...}` commands for thumbs generation:

- `\special{!thumb ...}`
- `\special{!dthumb ...}`

The first command affects only the current page; the second sets the default to be used on the rest of the document. Both `\special`'s have identical syntax and parameters.

There are also two ways the thumbs can be generated. You can specify the thumb image as an external file, or ask VTEX to generate a low-res page approximation dynamically. The first option is selected with the `f=` parameter in the `\special{...}` syntax; all the other parameters select the second option.

The valid parameters are:

f= Select an image file. The image file must be a bitmapped graphics file (not `.eps`). This option is mutually exclusive with the rest.

w= Define (in pixels) the width of the thumb to generate.

h= Define (in pixels) the height of the thumb to generate.

c= Define (hex, rrggbb) the color of the thumb.

b= Define (hex, rrggbb) the bacground color of the thumb.

Multiple parameters are separated with a comma. Examples:

- `\special{!thumb f=pic.gif}` loads the thumb for the current page from the `pic.gif` file.
- `\special{!thumb h=110,w=85}` creates 110×85 black and white thumb for the current page.
- `\special{!dthumb h=110,w=85,c=ff0000,b=00ff00}` creates 110×85 red-on-green thumbs for all subsequent pages.

Notes:

- specifying no parameters disables thumbs either for the current page only (`\special{!thumb}`) or for all subsequent pages (`\special{!dthumb}`).
- The global default is no thumbs.
- After a (`\special{!thumb...}`) the thumb settings revert to those given by the preceding (`\special{!dthumb...}`), or, if there were not one, to the no-thumbs default.
- Acrobat rescales the thumbs in all cases. Thus, only the proportion between the w= and h= settings matters; doubling the values of w= and h= will not change the thumbs appearance except for a small improvement in quality.

3.10 Annotations

The information in this section applies to the PDF mode only.

VTEX allows to place text annotations into the PDF file. This could be quite useful if you are working on a document with a collegue: text annotations could be used as “internal notes” to be deleted later.

This note has been produced with:

```
\verb+\special{!annotate w=5cm, h=4cm, t=For exa...}+.
```

The parameters understood by this `\special` are:

t= Annotation text, should always be the last parameter.

w= Open annotation width (`dimen`).

h= Open annotation height (`dimen`).

a= Annotation attributes (enclose in `<...>`, see PDF reference for additional details).

For example, use `a=</C [1 0 0]>` to have t[illegible]nnotation appear in red, rather than the default yellow, for example

```
\special{!annotate a=</C [1 0 0]>,t=like this}
```

The text within annotations may contain embedded `\char` commands which are resolved as explained in section 3.7.

3.11 Non-Latin characters in Outlines and Annotations

Acrobat 4 allows to put non-Latin characters into outlines and annotations. These characters are coded in Unicode. To simplify the task of entering such text, VTEX 6.5+ adds a couple of new primitives:

- `\unicode` is a 256-character table which defines the Unicode values for all characters. This table behaves in a way similar to the `\catcode` or `\lccode` tables.
- `\unithe` expands tokens into Unicode double-tokens, similarly to the usual `\the` operator.

Equipped with these operators, we can easily enter non-Latin text into annotations.

Similar technique works with outlines. Notice that `\char254\char255` is the usual Unicode text prefix marker; it must be supplied to inform the Adobe Reader that the following text is indeed in Unicode.

☞ Unicode text in annotations/outlines will be visible only on computers that have the required system fonts. In Win95/98/NT this specifically means installing the multilingual support options of the operating system.

3.12 PDF information special

The `\special{!pdfinfo...}` command is used to set the general PDF display and information options. The tail of the command can contain one or more of the following tags:

`a=<...>` Define the Author of the document.
`t=<...>` Define the Title of the document.
`s=<...>` Define the Subject of the document.
`k=<...>` Define the Keywords of the document.
`p=<...>` Define the Page Mode of the document.

The Author, Title, Subject and Keywords fields can be viewed in the Acrobat Reader by going into the **[File]/[Document Info]** selection.

The Page Mode field defines how the document should be opened by the Acrobat; notice that specifying illegal keys for Page Mode may lead to Acrobat crashes:

`/UseOutlines` means that the document should be opened with the outline visible.
`/UseThumbs` means that the document should be opened with the thumbnails visible.
`/FullScreen` means that the document should be opened in the full screen mode.

If you need to set more than one display option, use multiple `\special`'s.

In the PostScript mode this information is emitted as `pdfmark`'s.

☞ With LaTeX, we recommend to use the generic interface provided by the `hyperref` package instead of the low-level `\special{!pdfinfo...}` command.

3.13 PDF security settings

PDF security settings can be specified on the command line as the argument of the option `-e`, i.e.

`-e`*settings*

or through a `\special` command:

`\immediate\special{!security` *settings*`}`

The *settings* are a string of key-value pairs, separated by commas (not spaces!):

`U=`*password*`,O=`*password*`,P`*flag*`,C`*flag*`,A`*flag*`,M`*flag*

The meaning of the keys is:

`U` User password: the password to read ("open") the document.
`O` Owner password: the password to override all security settings or change them with Adobe Exchange. (In Acrobat terminology: "security password")
`P` Printing the document.
`C` Copying of text or graphics to the clipboard.

`A` Adding and changing text notes and AcroForm fields.

`M` Modifying the PDF document, other than by adding or changing text notes and AcroForm fields.

with

password A string, whose length should be limited to 32 characters. If there are spaces, commas or quotes in the string, it must be enclosed in doublequotes ("). If doublequotes are used in the password, double them.

flag + or –, to allow or deny the particular action.

It is not necessary to supply all 6 keys. Notice, however, that the default value of the P, C, A, and M keys is "deny", if unspecified.

If a security special is issued after any output has been generated, you will get a nasty error message. `\immediate` before this `\special` is therefore strongly advised, and issuing it as the first command in your document is a good idea.

Example:

```
\immediate\special{!security U="Aldous Huxley",O=Island,P+}
```

The password `Aldous␣Huxley` is reqired to read the document; printing will be allowed. Copying to the clipboard, adding annotations, modifying the document or changing the security settings is protected by the password `Island`.

☞ Specifying security settings on the command line is somewhat tricky with Unix: If there are passwords enclosed in doublequotes, the entire argument of `vtexlnx` or `vtexsun`, respectively, must *additionally* be enclosed in single quotes.

☞ With LaTeX we recommend using Heiko Oberdiek's macro package `pdfcrypt`, which provides a high-level key-value interface to the security settings. It is supplied with VTeX/Free.

3.14 PDF Page Labels

Acrobat 4+ supports the new *Page Label* feature. Page Labels affect the way pages are numbered in Acrobat Menus and Controls; they have no effect on the "printed page" portion of the document.

Page labels are useful if you are creating online documentation which uses a page sequence other than the default 1, 2, 3 . . .; for example, if some pages are numbered with roman numerals, and some with arabic digits. To have the Acrobat number the pages the same way in its menus, use the `\special{!pdfpagelabel` command:

```
\special{!pdfpagelabels
  /Nums [ 0 << /S /r >> % From page 0, roman (/r)
          3 << /S /D >> % From page 3, arabic digits (/D)
   ]
  }
```

Notice that unlike most other `\special`'s in VTEX, here the syntax is taken from the PDF 1.3 specifications verbatim. The five numbering styles supported by PDF 1.3 are

`/r` lower case roman numbers
`/R` upper case roman numbers
`/d` digits
`/a` lower case latin letters
`/A` upper case latin letters

Two additional optional keys that can be used within the page label specifications are the prefix (`/P`, to be followed by a string), and starting value for the range (`/St`, to be followed by in integer, default 1). The prefix specifies additional text to be appended to the left of the labels; the numbering starts with the starting value. For example:

```
\special{!pdfpagelabels
  /Nums [ 0 << /S /D /P (Intro-) >>
          5 << /S /D /St 6>>
        200 << /S /D /P (App-) /St 201 >>
   ]
  }
```

Here, all the numbering is done with digits, but the Intro-duction and the App-endix pages are marked so.

The `\special{!pdfpagelabels...}` command may appear anywhere in the document; if more than one is specified, the last specification is used. VTEX does not check the validity of the arguments.

This `special` has no effect on previewing under Acrobat 3 or GhostScript.

☞ If you are using LATEX, the task of constructing the Page Labels specifications can be given to `hyperref`; with other formats, you need to build the specification manually.

3.15 Space optimization

The information in this section applies to the PDF mode only.

8.02 VTEX 8.0+ can perform additional optimization of the PDF output using the special coding PDF allows for the space character (which normally does not appear in the TEX output.) This makes the PDF output more readable; it also may lead to substantial size reduction.

Space optimization is enabled by the `o` subswitch on the command line, see section 3.2.

For example, compiling the TEX Book without space optimization results in a file of 4947022 bytes; with space optimization the size drops to 4461703 bytes (uncompressed output). For fully compressed output the sizes are 2084239 (no space optimization) and 1982012 (space optimization).

Space optimization requires analyzing the encoding of the fonts before they are loaded. This may slow down the compilation, unless one used compiled

font maps, as explained in section 2.3.7). Currently the space optimization works only with Type 1 fonts.

3.16 Bitmapped graphics inclusion

Bitmapped graphics inclusion should be done through the LaTeX macro package `graphicx`; see the document `texmf/doc/latex/graphics/grfguide.pdf` for a general description of the package. This section explains, which graphics file formats are supported, how VTEX processes bitmapped files in the PDF and PS modes, and additional `\includegraphics` keys that can be of use in fine-tuning the performance. Further facilities for brightness, contrast and colorspace adjustment, as well as for image downsampling, are described in the GEX documentation.

3.16.1 Supported graphics formats

The VTEX/Free compiler features all of the bitmapped graphics filters also found in the windows version: **BMP** (i.e. Windows BMP), **GIF**, **JPEG**, **PCX**, **PNG**,
8.02 **TARGA** and **TIFF** images can all be included directly. As to JPEG, more modern types (i.e. progressive) will be included correctly, but Acrobat Reader 3 will not be able to show them right (gray rectangle only); Acrobat 4+ *will* show them.

7.47 When generating PDF output, one can also include **AVI** movies. Notice that the Acrobat Reader does not include a movie player and relies on an external tool. On Unix and OS/2 rendering of `.avi` movies in PDF documents is probably not (yet) supported at all.

3.16.2 Image size

The physical size of bitmapped graphics on the printed page can be specified in two ways:

- ▷ The desired height and/or width can be specified through the `height=` and `width=` keywords of the `\includgraphics` command. Upon rendering to an output device, the original bitmap will be scaled to these dimensions.
- ▷ Use the keyword `atres=`*resolution*. VTEX will then determine the physical dimensions of the image from its size in pixels and the indicated resolution. When printed at this very resolution, the bitmap needs not to be scaled; when the resolution of the output device is different, the bitmap will be scaled appropriately.

3.16.3 Compression and fine-tuning

Few ground rules to start with:

1. VTEX supplies two distinct techniques for loading the bitmapped files: direct processing and loading into the integrated PostScript interpreter,

GEX. In many cases the results are identical, but there are several differences, explained below. The default behavior is to process the pictures directly: this is always faster. However, there are some advantages in going through GEX: for example, you can post-process the images with the GEX imaging plugins (see the GEX documentation).

To load a bitmapped image for GEX inclusion processing, use the `viagex` key for the `\includegraphics` command.

2. The default processing logic for the direct picture inclusion is to try to preserve the structure of the image as it comes from the original file. For JPEG images, it means graphics inclusion of DCT-compressed data, exactly as it was stored in the source file; for other images it means copying the color space as specified in the original file. Assuming that the original file was optimally built, this logic produces optimal image in the PDF file as well as minimizes the operations on the image.

   Use the `repackjpeg` key for the `\includegraphics` command to force a JPEG image to be DCT-decompressed.

3. In order to force minimal size of directly included images, you can make VTEX apply CCITT or DCT compressions. For monochrome images, CCITT often produces much more compact results than the default Flate compression. For the color images, especially the photographs, DCT often improves on Flate, *even when DCT is lossless*. Note that CCITT and DCT compression are *not* controlled by the command line option `-$p(g=f`.

   The related keys for the `\includegraphics` command are `ccitt` and `dct`.

4. CCITT compression applies to monochrome images only, including imagemasks, and has no effect on color images. It may produce worse results on some pictures which are drawn rather than scanned (specifically, if they contain gray areas emulated with bit patterns).

5. The DCT compression by default is lossless; but often you do not mind losing some data, if this does not result in clear loss of quality and reduces the PDF size. The `dctquality=###` key for the `\includegraphics` command specifies the quality loss you are willing to sustain; with the default value of 0, there is no loss, the largest allowed value is 1000000 (loss of most of the data).

6. When using GEX(keyword `viagex`), the default processing logic for the graphics inclusion is to fully unpack the image. This means applying DCT-decompression to the JPEG files as well as de-indexing indexed graphics files. The rationale is make the image data easily accessible to the imaging plugins; the drawback is that in many cases this will cause larger output.

   You can force the image data to be re-indexed with the `reindex` key for the `\includegraphics` command.

Case study: monochrome GIF

The `herzjesu.gif` sample file is a monochrome `.gif` image; the original file size is 18396 bytes. This is one case when the default loading procedure will produce non-optimal results: GIF's are always built around a 256-color palette, even when only two colors (monochrome!) are needed. The default loading of this file, done with this code

```
\documentclass{article}
\pagestyle{empty}
\usepackage{graphicx}
\begin{document}
\includegraphics[width=1in]{herzjesu.gif}
\end{document}
```

will result in a 618k PDF file (uncompressed), or a 36k PDF file (flate-compressed). If the image were loaded via GEX, with

```
\includegraphics[width=1in,viagex]{herzjesu.gif}
```

the results will be even worse: 1815k (uncompressed), or 48k (Flate-compressed). However, repacking the image, accomplished with

```
\includegraphics[width=1in,viagex,reindex]{herzjesu.gif}
```

will bring the size down to 77k (uncompressed), or 9148 bytes (flate compressed).

☞ Reindexing can substantially reduce the size of monochrome GIF images, or any image that has been incorrectly packed.

Case study: Photo Bitmap

The `mac400.bmp` file is a photo bitmap. The default inclusion of this image file, done with

```
\pagestyle{empty}
\documentclass{article}
\usepackage{graphicx}
\begin{document}
\includegraphics{mac400.bmp}
\end{document}
```

will result in a 402k PDF file (Flate compression). However, the use of lossless DCT compression, as in

```
\includegraphics[dct]{mac400.bmp}
```

will result in only 369k file. With

```
\includegraphics[dct,dctquality=10]{mac400.bmp}
```

the size of the file will decrease to 221k, without much quality loss. Specifying `dctquality=100` makes the size fall to 70k, with some loss for printing, but not for the screen display; at `dctquality=1000` the file size will be down to 20k (with a very visible quality degradation).

☞ Use of DCT compression often decreases the size of the PDF files, even when

the compression is lossless. Notice that the `dct` key will have *no* effect on JPEG images which are already DCT compressed. If you would like to recompress/degrade JPEG images, use the key `repackjpeg` together with `dct` and `dctquality`.

3.16.4 Imagemasks

The `imagemask` keyword for the `\includegraphics` command applies to monochrome bitmapped images only. By default, images are not subjected to text color changes. With `imagemask` on, the images are treated as imagemasks: they assume the text color.

`imagemask` has a second application besides the ability to color images: when this keyword is specified, the image becomes *transparent*; this allows to overlap text and images without the risk of the image erasing the text.

3.17 PostScript file inclusion

Both the PDF and PostScript backend fully support inclusion of (encapsulated) PostScript files. The file type is recognized from their extension `.ps` or
8.02 `.eps`. From version 8 on, VTEX/Free can also read gzip-compressed PostScript files with the extension `.gz`, `ps.gz` or `eps.gz`.

In PDF mode, PostScript→PDF translation is performed by the GEX converter.

With the PostScript backend, PostScript files can be included without using GEX: they are simply passed unchanged to the `.ps` output. *With* GEX, however, the PS code is reinterpreted and optimized. One of the benefits of this is the combining of the fonts and other resources that are often repeated in included PostScript files.

You can use the LATEX packages `graphicx`, `graphics`, `epsfig` or `psfig` to include PostScript images. The latter is obsolete and is provided for use with legacy documents only. The others constitute different interfaces to the same macro code; see the document `texmf/doc/latex/graphics/grfguide.pdf` for the documentation. Of the four choices, `graphicx` is the one we recommend.

3.17.1 Page selection

Using `graphicx.sty`, a single page of a multi-page PostScript file can be selected for inclusion. Notice that this feature requires GEX. The `\includegraphics` command provides an additional key `page=` to specify the *absolute* page number within the file.

3.17.2 Font handling in included graphics files

By default, GEX will behave like a normal PostScript Level I or II interpreter and “knows” only the particular base fonts. Thus, inclusion of EPS files may

cause problems, if they refer to fonts which are neither base fonts nor embedded in the EPS file: GEX will simply replace any unknown font with Helvetica. GEX can, however, be told to recognize *all* Type 1 fonts, which have been installed for use with VTEX as explained in section 2.3, by issuing of the command

```
\special{pS: 1 .autofontload}
```

☞ in your TEX document. Notice that `.tfm` files for the raw (not reencoded) fonts must exist, even if they are not actually to be used by directly from TEX!

EPS files created by Adobe Illustrator may constitute a further problem: They may refer to the Monotype ArialMT and TimesNewRomanPSMT typefaces (base fonts of the Distiller) and include some tricky code to build ugly Type 3 equivalents, if the PostScript interpreter does not provide the fonts. Even if you install them, loading of the fonts through the above `\special` does not help, because the substitution occurs, before GEX will have looked up the fonts in the aliasing files. Instead, load the fonts explicitly through the following commands, before processing such EPS files:

```
\special{pS: /Arial-BoldItalicMT           .loadfont}
\special{pS: /Arial-BoldMT                 .loadfont}
\special{pS: /Arial-ItalicMT               .loadfont}
\special{pS: /ArialMT                      .loadfont}
\special{pS: /TimesNewRomanPS-BoldItalicMT .loadfont}
\special{pS: /TimesNewRomanPS-BoldMT       .loadfont}
\special{pS: /TimesNewRomanPS-ItalicMT     .loadfont}
\special{pS: /TimesNewRomanPSMT            .loadfont}
```

You may, of course, restrict yourself to those fonts which are actually needed. See also the sample file `tryAI.tex` in the `texmf/doc/vtex/examples` directory.

Base fonts in EPS graphics

EPS files may, in rare cases, include subsetted PostScript Base fonts, such as Times, Helvetica etc. When invoked with the "!" option (see section 3.2), VTEX may happen to use these incomplete fonts, rather than loading its own font files, which results later in missing characters. The only workaround is to call VTEX without the "!" option then. Notice that the `vlatex` and `vlatexp` shell scripts include the option, because it speeds up VTEX, and the problem is not very likely to occur.

3.17.3 Including METAPOST output

METAPOST output files (`.mps`) are valid EPS, provided that they have been generated with the METAPOST variable `prologues` set to `2`. METAPOST does *not* embed fonts; yet you need not issue the `autofontload` command as described above, since GEX will recognize the file type `.mps` and sets `autofontload` itself.

`.mps` files generated with `prologues=0` are, however, *not* valid PostScript. (Only the `dvips` program can handle them immediately.) Particularly, these files do *not* contain valid PostScript commands to load the fonts. In case you need to include this type of file into a TEX document and cannot re-make it using `prologues=2`, you may add some PostScript commands manually before the `\includegraphics` command(s) so as to enable proper processing. For instance, if the `.mps` file uses the fonts `cmr10`, `cmmi10` and `cmsy10`:

```
\special{pS: %
  /cmsy10 /cmsy10 def
  /cmmi10 /cmmi10 def
  /cmr10 /cmr10 def
  /fshow {exch findfont exch scalefont setfont show}bind def
  }
\includegraphics{file.mps}
```

3.18 Color stack issues

TEX color support is a later addition to TEX; Knuth's TEX kernel does not support color directly. As a result, the color commands do not obey either TEX syntax grouping or TEX boxes. In most cases, this is not a problem; but in some cases, the color may *leak* beyond what was intended, sometimes to the next page. To allow a macro designer to handle colors properly, VTEX now adds two more `\special{...}` commands:

- ▷ `\special{G(}`
- ▷ `\special{G)}`

The first of the commands is essentially equivalent to GEX's `\special{pS: gsave}`; the second is `\special{pS: grestore}`. However, these commands function even when GEX has not been enabled.

For TEX'nically minded: in VTEX, GEX's kernel is active even if GEX interpreter has not been enabled (i.e. no `-ox` switch given). It handles commands which deal with the colors and page transformations; thus some VTEX specials merely call GEX, whether it is enabled in full or not. Beside the two `\special`'s listed above, this applies to the rotation specials `\special{r(..}` and `\special{r)..}` as well as the color specials `\special{c...}`, both explained above. When the `-ox` switch given, some of these commands (`\special{r)...}`, for example) trigger full GEX initialization.

4 Language extensions

This chapter very briefly summarizes a few important syntax extensions in VTEX. A comprehensive description can be found in the VTEX Language Guide, which is currently provided with the commercial Windows version only.

- `\font` command: additional tags like `slant`, `outline`, `charmap`, and others allow to alter font shape and encoding dynamically.
- `\aliasfont` primitive allows to define an alias to the another font (the latter being possibly modified using the extended `\font` syntax). See the file `texmf/vtex/config/latex.fid`, which uses `\aliasfont` and the extended `\font` syntax extensively.
- An extension to the `\parshape` primitive allows paragraphs with "holes". See the document `parshape.tex` for an example.
- The `\exec` and `\command` primitives provide interface to launching external programs:

  The `\exec` command takes two arguments; the first is the name (and the path, if required) of an application to invoke; the second is the argument string.

  `\command` takes a single argument which is passed to the `command.com` or `cmd.exe` shell. Since commands like `dir` or `pause` are not programs under Windows and OS/2 but are handled by the command processor, they must be invoked via `\command`. On Linux, however, `ls` is a program, so it can be used in `\exec{}`.
- Almost unlimited number of fonts and registers: The number of fonts and TeX registers (`\count`'s, `\dimen`'s, `\box`'es) in VTEX is limited by 65536 (256 in standard TeX). Thus, it is entirely legal to write `\count1000=10`. These registers are allocated "on use". You can trace register reallocation if you set `\tracingallocs` to a positive value. LATEX register allocation does, however, not use these extra registers, so under LATEX you may still run out of registers, unless you use your own allocation scheme.

5 Implementation of the LaTeX packages `color` and `graphicx`

5.1 Color support

The `color` package allows to specify the color in either the RGB space or the CMYK space. On the low level this corresponds to VTEX color `\special`'s

- ▷ `\special{c"rrggbb}`
- ▷ `\special{c:ccmmyykk}`

All letter pairs in the syntax notation above correspond to two hex digits that define a hex number in the range of 0 through 255. Dividing this number over 255 gives the appropriate color component. For example, `\special{c"FF0000}` defines the red color. This color notation is patterned over the syntax used in HTML.

The macro that converts the `color.sty` notation of the color components to VTEX's notation has been contributed by David Carlisle.

5.2 Text-box scaling and Rotation

These features are implemented via the new `\special{r...}` command. This command modifies the PDF Current Transformation Matrix (CTM) in the PDF backend or its `.dvi` analog in the DVI backend. There are two forms of the command:

- ▷ Save and modify the CTM:

  `\special{r(c11,c12,c21,c22,0,0}`

  This will push the original CTM matrix to the stack and multiply it with the

  $$\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}$$

- ▷ Restore the CTM:

  `\special{r)}`

  This will pop the previous CTM from the stack.

Notice that both the VTEX drivers and the `.pdf` backend assume that CTM transformations are properly nested. The Graphics state stack overflows and underflows are treated as fatal errors.

Examples:

Scaling To scale a box by the factor of 2, use

```
\special{r(2,0,0,2,0,0} ... box ... \special{r)}
```

Non-uniform scaling To scale the y-coordinate only by the factor of three, use

```
\special{r(1,0,0,3,0,0} ... box ... \special{r)}
```

Flip To flip a box along the x-axis, use

```
\special{r(-1,0,0,1,0,0} ... box ... \special{r)}
```

Rotation To rotate a box 30°, use

```
\special{r(0.866,0.5,-0.5,0.866,0,0} ... box ... \special{r)}
```

Notice here that $0.866 \approx \cos(30/180 \times \pi)$ and $0.5 = \sin(30/180 \times \pi)$.

The responsibility of allocating sufficient space for the transformed box lies with the macro package designer. This allocation is automatically handled by the `\scalebox{..}` and `\rotatebox{..}` commands of the `graphicx` package.

The current implementation will transform text and rules but not graphics images within the text box.

6 The configuration of the VTEX/Free distribution

6.1 The file system

6.1.1 The `texmf` directory tree

The directory structure of VTEX/Free complies with the "TDS" standard, see `CTAN:tds`. On Unix, the root directory of the `texmf` tree is

`/usr/local/vtex/texmf`

and on OS/2 it is

drive`:\texmf`

on the partition (*drive*) where you have installed VTEX/2.

In contrast to what the TDS specifications suggests, the binaries are not located in the `texmf` tree. On Unix, they reside in `/usr/local/vtex/`; on OS/2, they reside in *drive*`:\vtex`.

VTEX-specific configuration files reside in a particular directory `vtex/config`, which comes first in the search path. When both VTEX/Free and another TEX system use the same directory tree, this directory is "hidden" from the non-VTEX system.

Thus, the actual `vtex.ini` supplied with VTEX/Free on OS/2 is:

```
[Directories]
PGMDIR=c:\vtex\bin\
INCDIR=c:\texmf\vtex\config\;c:\texmf\fonts\map\vtex\;c:\texmf\tex\generic\+
FMTDIR=c:\vtex\fmt\
TMPDIR=c:\scratch\
TFMDIR=c:\texmf\fonts\tfm\+
VFSDIR=c:\texmf\fonts\vf\+
ENCDIR=c:\texmf\fonts\enc\dvips\
BIBDIR=c:\texmf\bibtex\bib\+
BSTDIR=c:\texmf\bibtex\bst\+
MSTDIR=c:\texmf\makeindex\+
[THIRDPARTY]
A4FIX=0
[FINCLUDE]
LATEX=c:\texmf\vtex\config\;c:\texmf\fonts\map\vtex\;
  c:\texmf\tex\latex\+;c:\texmf\tex\generic\+
PLAIN=c:\texmf\vtex\config\;c:\texmf\fonts\map\vtex\;
  c:\texmf\tex\plain\+;c:\texmf\tex\generic\+;c:\texmf\tex\latex\graphics\
[HugeTeX]
trie_size=120000
trie_op_size=20000
[COMPILER]
RETCODES=0,0,1,1
```

(The records for LATEX and PLAIN are actually one single line each.) Note that the directory for the font mapping files (fonts/map/vtex) is listed in the path of the "TEX input files", because this is where VTEX searches for these files.

6.1.2 User-specific search paths

Many TEX systems support several texmf directory trees. Particularly in multi-user environments TEX has usually one or more system-wide directory trees and a user-specific one. With VTEX on Unix, you may keep vtex.ini in your $HOME directory, rather than in /etc, and specify your "personal" paths there. See also section 2.2.

There are, however, two important restrictions as to user-specific search paths:

- ▷ A fast static cache (-pr or -pr0) cannot be used: If the cache is under the control of the system maintainer, it would not reflect changes to your personal directories, and vice versa.
- ▷ A further restriction affects font files. In order to ensure a consistent setup, VTEX reads only one single .fm file. You would have to keep it in the "personal" TDS tree, if you want it to load any "personal" font mapping files, but it may get out of sync with the system-wide setup then. Thus, VTEX does not support to keep fonts in independently maintained directory trees.

Yet, having a user-specific vtex.ini does makes sense. You can still specify "personal" search paths for TEX macro files by adding them to the the LATEX and PLAIN variables. Directories for BibTeX style files (.bst) are to be added to BSTDIR, paths for BibTeX database files(.bib) are to be added to BIBDIR, and the variable MSTDIR is for MakeIndex style file (.ist). Note that you must not use environment variables to specify directories here; VTEX can, unfortunately, not evaluate them.

6.2 Font usage

6.2.1 FM and aliasing files

The VTEX/Free distribution includes shell scripts to run VTEX with the plain or LATEX formats in PDF or PostScript mode – see section 2.1. These shell scripts specify the usage of the FM files pdf.fm for PDF mode and ps.fm for PostScript mode:

```
% file pdf.fm for VTeX/Free -- PDF mode
LOCAL-OS2 {
  TEXMF = "c:/texmf/"
  }
LOCAL-LNX {
  TEXMF = "/usr/local/vtex/texmf/"
  }
```

```
PSRENAME {
  fontname.fix
  }
TYPE1 {
  tm-eu.ali
  hv-eu.ali
  pdfbase.ali
  cm.ali
  misc.ali
  micropress.ali
  cmbright.ali
  concrete.ali
  }
```

`ps.fm` differs from the above `pdf.fm` only with respect to the mapping files for the base fonts – see below.

The local variable `TEXMF` *must* be defined, because the aliasing files will refer to it. `TEXMF` must always point to the root directory of the TDS tree. A further variable `URW` may be present in FM files from older VTEX versions and is no longer used. Each aliasing file (font mapping file) refers to a certain set of fonts:

`tm-eu.ali`	Times
`hv-eu.ali`	Helvetica
`pdfbase.ali`	the remaining 27 base fonts
`cm.ali`	Computer Modern and related fonts
`misc.ali`	various free fonts that come wit VTEX/Free
`micropress.ali`	MicroPress' TM-Math, HV-Math
`cmbright.ali`	CM Bright
`concrete.ali`	Concrete

8.02 The aliasing files specify the use of URW's "clones" of the PostScript base fonts, which are shipped with VTEX/Free from release 8.02 on.

6.2.2 Embedding of the PostScript base fonts

Using `pdf.fm`, i.e., the shell script for **PDF** mode, all fonts are embedded as subsets into the generated files, With the shell scripts for **PostScript** mode, in contrast, the FM file `ps.fm` specifies a different set of aliasing files for the base fonts:

`tm-x.ali`	for Times
`hv-x.ali`	for Helvetica
`posbase.ali`	for the remaining base fonts

Using these mapping files, VTEX will *not* embed the base fonts. Under normal circumstances this is appropriate for PostScript output.

6.2.3 Additional fonts, which are prepared in VTEX/Free

Adobe Euro

VTEX/Free comes with a macro package, the font metrics and a font map file for the "Adobe Euro" symbol fonts. Due to legal restrictions, you have to download the fonts from Adobe's ftp server; they are *not* distributed in conjunction with VTEX. The fonts are available for free, and installing them is highly recommended.

Download the file

`ftp://ftp.adobe.com/pub/adobe/type/win/all/eurofont.exe`

or

`ftp://ftp-pac.adobe.com/pub/adobe/type/win/all/eurofont.exe`.

This is a self-extracting archive for DOS/Win, which can also be unpacked using *unzip*. Copy all files of type `.pfb` from the subdirectory `eurofont` of the archive to the directory `texmf/fonts/type1/adobe/eurofont` of your VTEX system, and rename them to "standard" file names:

Adobe file name:	standard file name:
`_1_____.PFB`	`zpeurs.pfb`
`_1B____.PFB`	`zpeubs.pfb`
`_1I____.PFB`	`zpeuris.pfb`
`_1BI___.PFB`	`zpeubis.pfb`
`_2_____.PFB`	`zpeurt.pfb`
`_2B____.PFB`	`zpeubt.pfb`
`_2I____.PFB`	`zpeurit.pfb`
`_2BI___.PFB`	`zpeubit.pfb`
`_3_____.PFB`	`zpeur.pfb`
`_3B____.PFB`	`zpeub.pfb`
`_3I____.PFB`	`zpeuri.pfb`
`_3BI___.PFB`	`zpeubi.pfb`

Use of the fonts is described in the document `texmf/doc/latex/eurosans/README.eurosans`.

Adobe Helvetica

While most of URW's clones of the PostScript base fonts are perfectly equivalent to Adobe's originals, Nimbus Sans exhibits a minor bug with the German sharp s (ß). Thus, we recommend to install the original Helvetica fonts and make VTEX use them. The following fonts files are required:

FontName:	Adobe file name:	standard file name:
Helvetica	`hv_____.pfb`	`phvr8a.pfb`
Helvetica-Bold	`hvb____.pfb`	`phvb8a.pfb`
Helvetica-Oblique	`hvo____.pfb`	`phvro8a.pfb`
Helvetica-BoldOblique	`hvbo___.pfb`	`phvbo8a.pfb`

You can either purchase them from Adobe or take them out of a product which includes them, e.g., Acrobat 3. Rename the font files (`.pfb`) to

the “standard” names given above and copy them to the directory texmf/fonts/type1/adobe/helvetic of your VTEX system. Edit the file texmf/vtex/config/pdf.fm and replace (in the Type1 section) hv-eu.ali with hv-eak.ali. (If you are too lazy to rename the font files, or if you have installed the font files with Adobe names already, use hv-ea.ali instead. Just make sure that the file names are lower case.)

Alternatively, if you have got the MicroPress HV-Math font set, you can use MicroPress' Helvetica “clones”, which exhibit the same quality; see below.

MicroPress CM-Bright, Concrete, TM-Math, HV-Math

VTEX/Free already includes the metric, macro and and font mapping files for the popular Concrete, CM-Bright, TM-Math and HV-Math typefaces. The required Type 1 font files (.pfb) are, however, *not* part of the distribution; they can be purchased from MicroPress Inc <http://www.micropress-inc.com/fonts>. Note that you need the “non-VTeX” versions of the particular distributions, since the variants for “VTeX users” are packaged for use with VTEX for Windows only!

Having purchased these fonts, all you need to do is to to copy the .pfb font files into the following directories:

CM-Bright	texmf/fonts/type/micropress/cmbright
Concrete	texmf/fonts/type/micropress/concrete
TM-Math	texmf/fonts/type/micropress/tmmath
HV-Math	texmf/fonts/type/micropress/tmmath

With the HV-Math fonts, it is recommended to make VTEX embed the Helvetica “clones”, which are part of the HV-Math bundle, rather than URW Nimbus Sans: Edit the file texmf/vtex/config/pdf.fm and replace (in the Type1 section) hv-eu.ali with hv-em.ali. Or, if you have got Adobe's originals, use these, as described in the above section.

MicroPress BA-Math, Informal-Math, PA-Math

VTEX/Free includes only the required font mapping files. Follow the installation instructions that come with the fonts, and add an entry for the particular mapping file to the TYPE1 section of the configuration files texmf/vtex/config/pdf.fm and ./ps.fm, for instance

```
TYPE1 {
  ...
  bam-em.ali
}
```

for BA-Math. The corresponding file names for the Informal-Math and PA-Math fonts are if1.ali and pam-em.ali.

Monotype Arial and TimesNewRoman

Certain programs use the font families Monotype ArialMT and TimesNewRomanPSMT as a replacement for the Helvetica and Times typefaces.

As to TEX, ArialMT and TimesNewRomanPSMT should *not* be used explicitly. Yet VTEX's built-in PostScript interpreter may need them in order to process EPS graphics, which refer to the fonts but do not include them; see section 3.17.2. This applies particularly to EPS graphics created by Adobe Illustrator.

Assuming you have the 8 needed fonts (they are part of the Acrobat 4 and Acrobat 5 distributions or can be ordered separately from Adobe), just copy or link the following Type 1 font files to the directory `texmf/fonts/type1/monotype/acrobat` of your VTEX system:

```
_abi_____.pfb
_ab______.pfb
_ai______.pfb
_a_______.pfb
_ebi_____.pfb
_eb______.pfb
_ei______.pfb
_er______.pfb
```

Not installing the fonts will not cause any problems, unless you actually encounter a graphics file that needs them.

7 Support

The NTG hosts a mailing list that can be used both to ask questions and to report bugs in the software and/or installation. New versions of VTEX/Free will also be announced there. The list has a web interface for subscribing, unsubscribing, accessing the archives etc:

`http://ref.ntg.nl/mailman/listinfo/ntg-vtex`

If you are using the software, being on the maillist is a very good idea. This is the way to be informed about the newest features, about problems others encountered (and how they were solved), it is also the way to voice your opinion about our development and have it count.

Your feedback is very valuable to us. We have spent a plenty of efforts in making this software and the distribution and we need your feedback to know how to make it yet better. If you encounter a bug, or a inconsistency in the documentation, or if you see a way to make things work smoother, please share it with us on the maillist.

8 License Terms

MicroPress and VTEX are trademarks of MicroPress, Inc.

The MicroPress' components of the VTEX/Free distribution are

> Copyright © 1998–2003 by MicroPress Inc.

The MicroPress' components of the VTEX/Free distribution are free for personal use, subject to the following restriction:

You must obtain a permission from either MicroPress Inc or Walter Schmidt, if you intend to redistribute VTEX/Free on a permanent media (CD), or host it on a Web site other than by mirroring the official distribution.

The VTEX/Free distribution includes components that are covered by different licenses, as follows:

The Type 1 fonts in the subdirectories of `texmf/fonts/type1/urw` are distributed under the GNU License, see the file `GPL`. Each of these fonts is individually covered by the license: for licensing purposes, they are not "part of" any larger entity. The following notice applies to these fonts:

> Copyright (URW)++,Copyright 1999 by (URW)++ Design & Development

The Adobe Utopia fonts in the directory `texmf/fonts/type1/adobe/utopia` are subject to the following conditions of use and distribution:

> Permission to use, reproduce, display and distribute the listed typefaces is hereby granted, provided that the Adobe Copyright notice appears in all whole and partial copies of the software and that the following trademark symbol and attribution appear in all unmodified copies of the software:
>
> Copyright © 1989 Adobe Systems Incorporated.
>
> Utopia® is a registered trademark of Adobe Systems Incorporated.

The Bitstream Charter fonts in the directory `texmf/fonts/type1/bitstrea/charter` are subject to the following conditions of use and distribution:

> © Copyright 1989–1992, Bitstream Inc., Cambridge, MA.
>
> You are hereby granted permission under all Bitstream propriety rights to use, copy, modify, sublicense, sell, and redistribute the 4 Bitstream Charter® Type 1 outline fonts for any purpose and without restriction; provided, that this notice is left intact on all copies of such fonts and that Bitstream's trademark is acknowledged as shown below on all unmodified copies of the 4 Charter Type 1 fonts:
>
> Bitstream Charter is a registered trademark of Bitstream Inc.

Notice also the license conditions in the files of the LATEX system!