



User's Guide

Version 0.7.1

Gerd Neugebauer

This document describes $\epsilon x\text{TeX}$. It explains how to get $\epsilon x\text{TeX}$ up and running and which features $\epsilon x\text{TeX}$ offers to you. Since $\epsilon x\text{TeX}$ provides a testbed for experimentation the focus has been put on the default configurations. The intended audience for this document are end users of the typesetting engine who want to use $\epsilon x\text{TeX}$ on the command line or as plug-in replacement of TeX .

© 2005 The $\varepsilon\chi$ TeX Group and individual authors listed below

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation. A copy of the license is included in the section entitled “GNU Free Documentation License”.

Gerd Neugebauer
Im Lerchelsbühl 5
64521 Groß-Gerau (Germany)
gene@gerd-neugebauer.de

Contents

1. Introduction	5
1.1. This Document	5
1.2. Web Site	5
1.3. Mailing Lists	5
1.4. Reporting Bugs	5
2. Getting Started	7
2.1. Prerequisites	7
2.1.1. Java	7
2.1.2. TEXMF	7
2.2. Getting $\epsilon\lambda\text{TeX}$	8
2.2.1. Getting the Installer	8
2.2.2. Getting the Sources	8
2.3. Installing $\epsilon\lambda\text{TeX}$	9
2.3.1. Installing $\epsilon\lambda\text{TeX}$ with the Installer	9
2.3.2. Replaying an Installation	10
2.3.3. Creating the $\epsilon\lambda\text{TeX}$ Installer	10
2.3.4. Installing $\epsilon\lambda\text{TeX}$ from the Sources on the Command Line	11
2.4. Configuring $\epsilon\lambda\text{TeX}$	11
2.4.1. Startup Files	11
2.4.2. Configuration Files	16
2.4.3. Predefined Configurations	16
2.4.4. Primitive Sets	17
2.5. Running $\epsilon\lambda\text{TeX}$	19
2.5.1. Command Line Parameters	20
2.5.2. Creating Formats	24
3. Troubleshooting $\epsilon\lambda\text{TeX}$	25
3.1. Why are my files not found?	25
3.2. Why are is the log file different from TeX 's?	25
4. The Macro Language of $\epsilon\lambda\text{TeX}$	27
4.1. Primitives of $\epsilon\lambda\text{TeX}$	27
4.2. Basic Syntactic Entities of $\epsilon\lambda\text{TeX}$	122

A. Licenses	123
A.1. GNU Free Documentation License	123
A.2. GNU Library General Public License	125

1. Introduction

$\epsilon\chi\text{T}_{\text{E}}\text{X}$ aims at providing a high-quality typesetting system. The development of $\epsilon\chi\text{T}_{\text{E}}\text{X}$ has been inspired by the experiences with $\text{T}_{\text{E}}\text{X}$. The focus lies on an open design and a high degree of configurability. Thus $\epsilon\chi\text{T}_{\text{E}}\text{X}$ should be a good base for further development.

On the other hand we have to take care not to leave the current user base of $\text{T}_{\text{E}}\text{X}$ behind. $\text{pdfT}_{\text{E}}\text{X}$ has taught us that a migration path from $\text{T}_{\text{E}}\text{X}$ has a positive value in it. In the mean time the majority of $\text{T}_{\text{E}}\text{X}$ users applies in fact $\text{pdfT}_{\text{E}}\text{X}$.

To provide a backward compatibility of $\epsilon\chi\text{T}_{\text{E}}\text{X}$ with $\text{T}_{\text{E}}\text{X}$ one special configuration is provided. Thus backward compatibility is just a matter of configuration.

1.1. This Document

This document is meant to be a reference document. It should contain all information necessary to know. It is not meant to be a tutorial. Thus do not expect tutorial type material in this document.

1.2. Web Site

There is a web site devoted to $\epsilon\chi\text{T}_{\text{E}}\text{X}$. This web site can be reached via the URL

<http://www.extex.org>

1.3. Mailing Lists

If you are ready to try $\epsilon\chi\text{T}_{\text{E}}\text{X}$ you might as well want to join a mailing list to get in contact with the community.

<http://www.dante.de/listman/extex>

1.4. Reporting Bugs

If you find any bugs in $\epsilon\chi\text{T}_{\text{E}}\text{X}$ you can submit them either via a HTML form or via email. You can find the HTML form at

<http://www.extex.org/bugs>

1. Introduction

Emails containing the description can be sent to

extex-bugs@dante.de

Please include in your description

- the source of a *minimal* example showing the problem
- the log file resulting from running this example
- a description why you think that something went wrong and what the expected result would be
- a description of the environment you are using (host architecture, operating system, Java version)

2. Getting Started

In this chapter we describe the steps you can take to get $\varepsilon\chi\text{T}_{\text{E}}\text{X}$ up and running. We try to use as few as possible premises. Thus it should be not too hard to get started.

2.1. Prerequisites

2.1.1. Java

You need to have Java 1.4.2 or later installed on your system. You can get Java for a several systems directly from java.sun.com. Download and install it according to the installation instructions for your environment.

To check that you have an appropriate Java on your path you can use the command `java` with the argument `-version`. This can be seen in the following listing:

```
# java -version
java version "1.4.2_06"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.4.2_06-b03)
Java HotSpot(TM) Client VM (build 1.4.2_06-b03, mixed mode)
#
```

2.1.2. TEXMF

If you want to use more than the pure $\varepsilon\chi\text{T}_{\text{E}}\text{X}$ engine, fonts and macros can be inherited from a texmf tree. $\varepsilon\chi\text{T}_{\text{E}}\text{X}$ itself does not contain a full texmf tree. It comes just with some rudimentary files necessary for testing. Thus you should have installed a texmf tree, e.g. from a $\text{T}_{\text{E}}\text{X}$ Live installation. This can be found on the [Comprehensive \$\text{T}_{\text{E}}\text{X}\$ Archive Network \(CTAN\)](http://www.ctan.org).

There is no need to install the texmf tree in a special place. You have to tell $\varepsilon\chi\text{T}_{\text{E}}\text{X}$ anyhow where it can be found. It is even possible to work with several texmf trees.

One requirement for the texmf trees is that they have a file database (`ls-R`). $\varepsilon\chi\text{T}_{\text{E}}\text{X}$ can be configured to work without it, but then $\varepsilon\chi\text{T}_{\text{E}}\text{X}$ is deadly slow. Thus you do not really want to try this alternative.

2.2. Getting $\epsilon\chi$ TeX

2.2.1. Getting the Installer

The simplest way to get $\epsilon\chi$ TeX up and running is to use the $\epsilon\chi$ TeX installer. This installer is distributed as one file `ExTeX-setup.jar`. You can download it from

<http://www.extex.org/download/>

To be completed.

2.2.2. Getting the Sources

The sources of $\epsilon\chi$ TeX are stored in a CVS repository. To access this repository you need access to the internet and CVS installed in some way.

The coordinates of the repository are:

Connection type:	pserver
User:	anonymous
Host:	cvs.extex.berlios.de
Location:	/cvsroot/extex
Module:	ExTeX

We assume here that you have access to CVS on the command line. This can be either a shell on a Unix-like system or something like cygwin on Windows. We also assume that you have direct connection to the internet.

First we create a directory where the sources are stored:

```
# mkdir ExTeX
```

Next we change the current directory to this base directory:

```
# cd ExTeX
```

Now we log into the CVS repository. This login uses an anonymous account. This enables us to download the sources but not to commit any changes. The committing is restricted to members of the $\epsilon\chi$ TeX team.

```
# cvs -d:pserver:anonymous@cvs.extex.berlios.de/cvsroot/extex login
```

Finally we can check out the sources:

```
# cvs -d:pserver:anonymous@cvs.extex.berlios.de/cvsroot/extex co ExTeX
```

This command shows a lot of output. At the end the current directory is filled with a lot of files and directories.

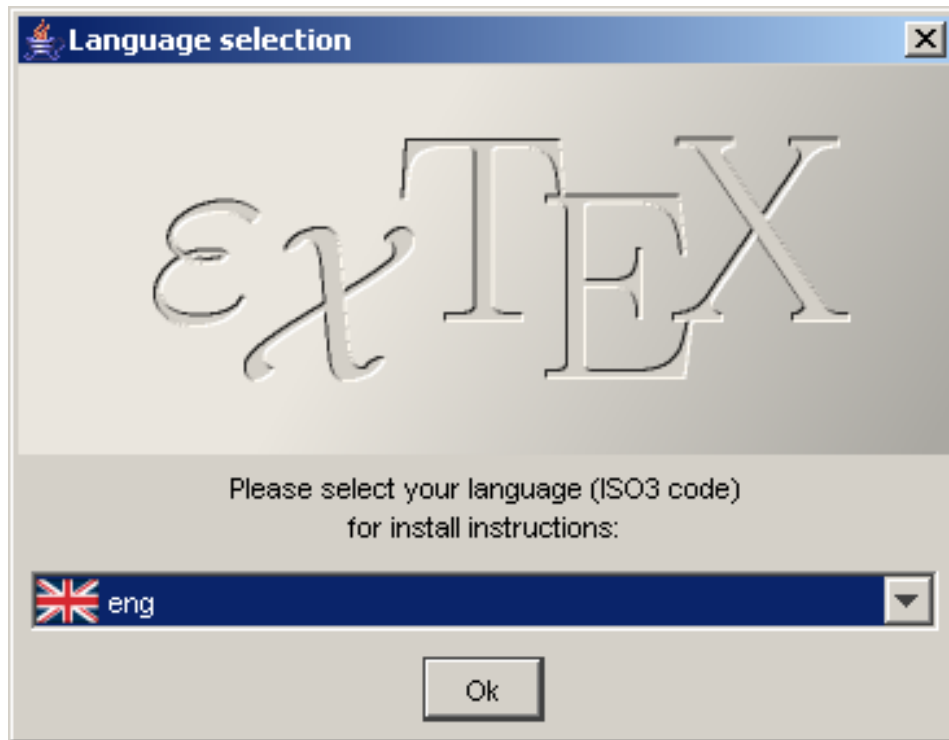


Figure 2.1.: The Language Selection in the Installer

2.3. Installing $\epsilon\chi\text{TeX}$

There are several ways to install $\epsilon\chi\text{TeX}$. Some of them are described in this section.

2.3.1. Installing $\epsilon\chi\text{TeX}$ with the Installer

The easiest installation of $\epsilon\chi\text{TeX}$ works with the $\epsilon\chi\text{TeX}$ installer. This installer is named `ExTeX-setup.jar`. You can start the installer with the following command line:

```
# java -jar ExTeX-setup.jar
```

On Windows with a properly installed Java you can also start the installer by double-clicking `ExTeX-setup.jar` in the Explorer.

The installer provides a graphical user interface with a wizard guiding you through the installation process. The first dialog is shown in figure 2.1. As you can see you can select one of several languages for the installation process. Currently the languages English and German are supported. There might be some more at the time you are performing the installation.

Note that the internationalization covers the installer only. $\epsilon\chi\text{TeX}$ can be run under different language environments as well. This is controlled by a setting at run-time. Currently only an English language binding for $\epsilon\chi\text{TeX}$ is provided.

Finally you have to make sure that the executables `extex` or `extex.bat` is on your path for executables.

2. Getting Started

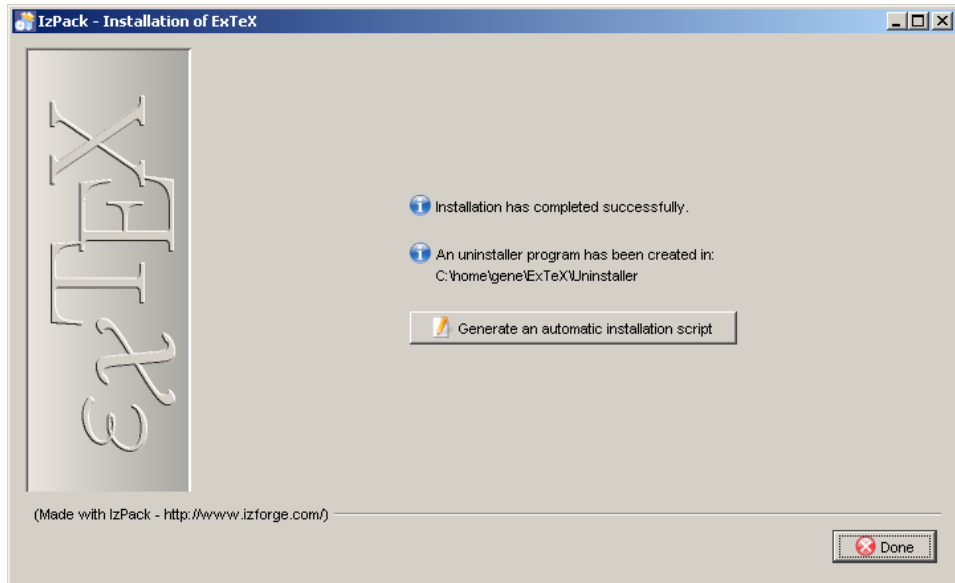


Figure 2.2.: Generating a Auto-Configuration for the Installer

2.3.2. Replaying an Installation

Sometimes it is desirable to perform an installation on several similar machines. This means that the answers to the questions in the installer are the same. This process can be automated.

In figure 2.2 you can see the last screen of the installer. Here you have the possibility to select the button “Generate an automatic installation script”. This produces an XML file which can be passed to the installer to avoid the dialogs.

Suppose you have named the file `replay.xml` in the file selector which pops up when the button has been pressed. Then you can replay the installation with the following command invocation:

```
# java -jar ExTeX-setup.jar replay.xml
```

This supposes that the two files `ExTeX-setup.jar` and `replay.xml` are in the current directory.

Finally you have to make sure that the executables `extex` or `extex.bat` is on your path for executables.

2.3.3. Creating the $\epsilon\chi\text{T}_{\text{E}}\text{X}$ Installer

You can create the installer of $\epsilon\chi\text{T}_{\text{E}}\text{X}$ from the sources. All you need for this step is contained in the source distribution. Suppose you are in the base directory of the distribution. Then the following command creates the installer:

```
# build installer
```

As a result the file `ExTeX-setup.jar` is created in the directory `target`. This file is a self-contained installer. You can immediately start the installer with the following command line:

```
# java -jar target/ExTeX-setup.jar
```

In addition the installer file can be moved to any other place – even other machines – and run the installation there (see also section 2.3.1).

2.3.4. Installing $\epsilon\chi\text{T}_{\text{E}}\text{X}$ from the Sources on the Command Line

To install you can use the build script provided in the $\epsilon\chi\text{T}_{\text{E}}\text{X}$ base directory.

```
# build -Dinstall.dir=/usr/local/share/ExTeX install
```

Additionally you have to copy the file `.extex` from the base directory of the $\epsilon\chi\text{T}_{\text{E}}\text{X}$ to your home directory and adapted to your installation. Most probably the value of the property `extex.texinputs` needs adaption to point to your texmf trees.

Finally you have to make sure that the executables `extex` or `extex.bat` is on your path for executables.

Now you can forget the source directory. It is not needed any more unless you are debugging or developing $\epsilon\chi\text{T}_{\text{E}}\text{X}$ extensions.

2.4. Configuring $\epsilon\chi\text{T}_{\text{E}}\text{X}$

The behaviour of $\epsilon\chi\text{T}_{\text{E}}\text{X}$ can be influenced via command line arguments and configuration files. Most of the times the startup files will be enough for the casual user.

2.4.1. Startup Files

Whenever $\epsilon\chi\text{T}_{\text{E}}\text{X}$ starts it looks for startup files named `.extex`. This file is sought in the user's home directory in in the current directory. The settings in the current directoy overwrite the settings from the user's home directory. Those in turn overwrite the built-in settings.

$\epsilon\chi\text{T}_{\text{E}}\text{X}$ user properties files contain setting of properties. This is done in a line-based way. Lines containing only whitespace characters are ignored. If the first character is a hash sign (#) then the line is treated as a comment and ignored.

The first appearance of a equal sign (=) or the colon (:) separates the name of the property from the value. Leading and trailing whitespace is ignored both for the name and the value of the property.

Some characters have a special meaning. The backslash (\) acts as an escape character. The sequence `\n` is replaced by the newline character. If the last character in a line is a backslash then the line is continued in the next line. To produce a single backslash it has to be doubled.

2. Getting Started

You can set any property name you like to a legal value. $\epsilon\chi\text{TeX}$ will not complain about unknown properties but ignore them silently. The following properties are used by $\epsilon\chi\text{TeX}$:

`extex.code`

This parameter contains $\epsilon\chi\text{TeX}$ code to be executed directly. The execution is performed after any code specified in an input file.

Example:

```
extex.code = \\relax
```

`extex.config`

This parameter contains the name of the configuration resource to use. This configuration resource is sought on the classpath.

Example:

```
extex.config = tex.xml
```

`extex.encoding`

This parameter contains the name of the property for the standard encoding to use.

Example:

```
extex.encoding = ISO-8859-1
```

`extex.error.handler`

This parameter contains the logical name of the error handler.

Example:

```
extex.error.handler = TeX
```

`extex.fonts`

This parameter contains the property indicating where to find font files. The value is a path similar to `extex.texinputs`.

Example:

```
extex.fonts = /usr/local/share/fonts
```

`extex.halt.on.error`

This boolean parameter contains the property indicating whether the processing should stop after the first error. Allowed values are `true` and `false`.

Example:

```
extex.halt.on.error = false
```

extex.file

This parameter contains the file to read from. It has no default. If this property is not set or set to the empty string then no attempt is made to read a file. Maybe the user is asked to provide one.

Example:

```
extex.file = abc.tex
```

extex.fmt

This parameter contains the name of the format to read. An empty string denotes that no format should be read. This is the default. In thi case $\epsilon\chi\text{T}_{\text{E}}\text{X}$ acts with no macros or fonts preloaded.

Example:

```
extex.fmt = plain
```

extex.ini

If set to `true` then act as `iniTEX`. This command line option is defined for compatibility to `TEX` only. In $\epsilon\chi\text{T}_{\text{E}}\text{X}$ it has no effect at all. Allowed values are `true` and `false`.

Example:

```
extex.ini = true
```

extex.interaction

This parameter contains the interaction mode. possible values are the numbers 0...3 and the symbolic names `batchmode` (0), `nonstopmode` (1), `scrollmode` (2), and `errorstopmode` (3).

Example:

```
extex.interaction = scrollmode
```

extex.jobname

This parameter contains the name of the job. It is overwritten if a file is given to read from. In this case the basename of the input file is used instead. If no file is read in then the default value `texput` is used.

Example:

```
extex.jobname = texput
```

extex.jobname.master

This parameter contains the name of the job to be used with high priority.

Example:

2. Getting Started

```
extex.jobname.master = texput
```

`extex.lang`

This parameter contains the name of the locale to be used for the messages. The value is a two letter ISO language code. $\epsilon\lambda\text{TeX}$ can be internationalized by providing some files with the translated strings. Currently only the language English (`en`) is supported.

Example:

```
extex.lang = en
```

`extex.nobanner`

This parameter contains a boolean indicating that the banner should be suppressed. Allowed values are `true` and `false`.

Example:

```
extex.nobanner = false
```

`extex.output`

This parameter contains the output format. This logical name is resolved via the configuration.

Example:

```
extex.output = pdf
```

`extex.outputdir`

This parameter contains the directory where output files should be created. The period is interpreted as the current directory. The default is the current directory.

Example:

```
extex.outputdir = .
```

`extex.outputdir.fallback`

This parameter contains the property for the fallback if the output directory (`extex.outputdir`) fails to be writable. The period is interpreted as the current directory.

The default is the current directory. Thus you can reset `extex.outputdir` and if this directory happens not to be writable then the current directory is used to create the log file and output files in.

Example:

```
extex.outputdir.fallback = .
```

extex.progname

This parameter can be used to overrule the name of the program shown in the banner and the version information.

Example:

```
extex.progname = iniExTeX
```

extex.stacktrace.on.internal.error

This parameter can be used to force a stacktrace on stdout if an internal error is encountered. This is handy for development. Allowed values are **true** and **false**.

Example:

```
extex.stacktrace.on.internal.error = true
```

extex.texinputs

This parameter contains the additional directories for searching $\epsilon\chi\text{T}_{\text{E}}\text{X}$ input files. The directories are separated by the system-dependant separator. This separator is a colon (:) on Unix and the semicolon (;) on Windows.

Example:

```
extex.texinputs = /home/gene/lib/tex
```

extex.trace.input.files

This boolean parameter contains the indicator whether or not to trace the search for input files. Allowed values are **true** and **false**.

Example:

```
extex.trace.input.files = false
```

extex.trace.font.files

This boolean parameter contains the indicator whether or not to trace the search for font files. Allowed values are **true** and **false**.

Example:

```
extex.trace.font.files = false
```

extex.trace.macros

This boolean parameter contains the indicator whether or not to trace the execution of macros. Allowed values are **true** and **false**.

Example:

```
extex.trace.macros = false
```

2. Getting Started

`extex.trace.tokenizer`

This boolean parameter contains the indicator whether or not to trace the work of the tokenizer. Allowed values are `true` and `false`.

Example:

```
extex.trace.tokenizer = false
```

`extex.typesetter`

This parameter contains the name of the typesetter to use. If it is not set then the default from the configuration file is used.

Example:

```
extex.typesetter = default
```

2.4.2. Configuration Files

Configuration files of another kind contain the assembly instructions for $\varepsilon\chi\text{T}_{\text{E}}\text{X}$. Those files can be used to provide additional features in $\varepsilon\chi\text{T}_{\text{E}}\text{X}$.

To be completed.

2.4.3. Predefined Configurations

The Configuration `extex`

The configuration `extex` identifies itself as “TeX compatibility mode”. The configuration contains the primitive sets `tex`, `etex`, and `omega`. The configuration allows extended register names.

The Configuration `extex-jx`

The configuration `extex-jx` identifies itself as “Java extensions”. The configuration contains the primitive sets `tex`, `etex`, and `jx`. The configuration allows extended register names.

The Configuration `extex-native`

The configuration `extex-native` identifies itself as “Native extensions”. The configuration contains the primitive sets `tex`, `etex`, and `native`. The configuration allows extended register names.

The Configuration nextex

The configuration `nextex` identifies itself as “Namespace extension”. The configuration contains the primitive sets `tex`, `etex`, and `namespace`. The configuration allows extended register names.

The Configuration omega

The configuration `omega` identifies itself as “Omega compatibility mode”. The configuration contains the primitive sets `tex`, `etex`, and `omega`.

The Configuration tex

The configuration `tex` identifies itself as “TeX compatibility mode”. The configuration contains the primitive set `tex`.

2.4.4. Primitive Sets**The Primitive Set etex**

The primitive set `etex` defines the following primitives:

```
\beginL \beginR \botmarks \clubpenalties \currentgrouplevel
\currentgrouptype \currentifbranch \currentiflevel \currentifttype
\detokenize \dimenexpr \displaywidowpenalties \endL \endR \eTeXrevision
\eTeXversion \everyeof \firstmarks \fontcharhp \fontcharht
\fontcharic \fontcharwd \glueexpr \glueshrink \glueshrinkorder
\gluestretch \gluestretchorder \ifcsname \ifdefined \interactionmode
\interlinepenalties \lastlinefit \lastnodetype \marks \middle \muexpr
\numexpr \pagediscards \parshapedimen \parshapeindent \parshapelength
\predisplaydirection \protected \readline \savingshyphcodes
\savingvdiscards \scantokens \showgroups \showtokens \splitbotmarks
\splitdiscards \splitfirstmarks \TeXXeTstate \topmarks \tracingassigns
\tracingcommands \tracinggroups \tracingifs \tracingnesting
\tracingscantokens \unexpanded \unless \widowpenalties
```

The Primitive Set jx

The primitive set `jx` defines the following primitives:

```
\javadef \javaload
```

The Primitive Set namespace

The primitive set `namespace` defines the following primitives:

```
\export \import \namespace
```

2. Getting Started

The Primitive Set `native`

The primitive set `native` defines the following primitives:

```
\nativedef \nativeload
```

The Primitive Set `omega`

The primitive set `omega` defines the following primitives:

```
\addafterocplist \addbeforeocplist \clearocplists \DefaultInputMode  
\DefaultInputTranslation \DefaultOutputMode \DefaultOutputTranslation  
\hfi \InputMode \InputTranslation \localbrokenpenalty  
\localinterlinepenalty \localleftbox \localrightbox \mathdir  
\naturaldir \noDefaultInputMode \noDefaultInputTranslation  
\noDefaultOutputMode \noDefaultOutputTranslation \nullocplist \ocp  
\ocplist \odelimiter \omathaccent \omathchar \omathchardef \omathcode  
\omathdelcode \oradical \OutputMode \OutputTranslation \pagedir  
\pagedirHL \pagedirHR \popocplist \pushocplist \removebeforeocplist  
\texdir \unnaturaldir \vfi
```

The Primitive Set `tex`

The primitive set `tex` defines the following primitives:

```
\_ \_ \_ \above \abovedisplayshortskip \abovedisplayskip  
\abovewithdelims \accent \adjdemerits \advance \afterassignment  
\aftergroup \atop \atopwithdelims \badness \baselineskip \batchmode  
\begingroup \belowdisplayshortskip \belowdisplayskip \binoppenalty  
\botmark \box \boxmaxdepth \brokenpenalty \catcode \char \chardef  
\cleaders \closein \closeout \clubpenalty \copy \count \countdef \cr  
\crrcr \csname \day \deadcycles \def \defaultthyphenchar \defaultskewchar  
\delcode \delimiter \delimiterfactor \delimitershortfall \dimen  
\dimendef \discretionary \displayindent \displaylimits \displaystyle  
\displaywidowpenalty \displaywidth \divide \doublehyphendemerits  
\dp \dump \edef \else \emergencystretch \end \endcsname \endgroup  
\endinput \endlinechar \eqno \errhelp \errmessage \errorcontextlines  
\errorstopmode \escapechar \everycr \everydisplay \everyhbox \everyjob  
\everymath \everypar \everyvbox \exhyphenpenalty \expandafter \fam \fi  
\finalhyphendemerits \firstmark \floatingpenalty \font \fontdimen  
\fontname \futurelet \gdef \global \globaldefs \halign \hangafter  
\hangindent \hbadness \hbox \hfil \hfill \hfilneg \hfuzz \hoffset  
\holdinginserts \hrule \hsize \hskip \hss \ht \hyphenation \hyphenchar  
\hyphenpenalty \if \ifcase \ifcat \ifdim \ifeof \iffalse \ifhbox  
\ifhmode \ifinner \ifmmode \ifnum \ifodd \iftrue \ifvbox \ifvmode  
\ifvoid \ifx \ignorespaces \immediate \indent \input \inputlineno  
\insert \insertpenalties \interlinepenalty \jobname \kern \language  
\lastbox \lastkern \lastpenalty \lastskip \lccode \leaders \left
```

```

\lefthyphenmin \leftskip \leqno \let \limits \linepenalty \lineskip
\lineskiplimit \long \looseness \lower \lowercase \mag \mark
\mathaccent \mathbin \mathchar \mathchardef \mathchoice \mathclose
\mathcode \mathinner \mathop \mathopen \mathord \mathpunct \mathrel
\mathsurround \maxdeadcycles \maxdepth \meaning \medmuskip \message
\mkern \month \moveleft \moveright \mskip \multiply \muskip \muskipdef
\newlinechar \noalign \noboundary \noexpand \noindent \nolimits
\nonscript \nonstopmode \nulldelimiterspace \nullfont \number
\omit \openin \openout \or \outer \output \outputpenalty \over
\overfullrule \overline \overwithdelims \pagedepth \pagefilllstretch
\pagefillstretch \pagefilstretch \pagegoal \pageshrink \pagestretch
\pagetotal \par \parfillskip \parindent \parshape \parskip
\patterns \pausing \penalty \postdisplaypenalty \predisplaypenalty
\predisplaysize \pretolerance \prevdepth \prevgraf \radical
\raise \read \relax \relpenalty \right \righthyphenmin \rightskip
\romannumeral \scriptfont \scriptscriptfont \scriptscriptstyle
\scriptspace \scriptstyle \scrollmode \setbox \setlanguage \sfcode
\shipout \show \showbox \showboxbreadth \showboxdepth \showlists
\showthe \skewchar \skip \skipdef \spacefactor \spaceskip \span
\special \splitbotmark \splitfirstmark \splitmaxdepth \splittopskip
\string \tabskip \textfont \textstyle \the \thickmuskip \thinmuskip
\time \toks \toksdef \tolerance \topmark \topskip \tracingcommands
\tracinglostchars \tracingmacros \tracingonline \tracingoutput
\tracingpages \tracingparagraphs \tracingrestores \tracingstats
\uccode \uchyph \underline \unhbox \unhcopy \unkern \unpenalty \unskip
\unvbox \unvcopy \uppercase \vadjust \valign \vbadness \vbox \vcenter
\vfil \vfill \vfilneg \vfuzz \voffset \vrule \vsize \vskip \vsplit \vss
\vtop \wd \widowpenalty \write \xdef \xleaders \xspaceskip \year

```

2.5. Running $\epsilon_X\text{T}_{\text{E}}\text{X}$

Currently $\epsilon_X\text{T}_{\text{E}}\text{X}$ can be run from the command line. In this respect it is more or less identical to $\text{T}_{\text{E}}\text{X}$ and can be used as a plug-in replacement.

The following sample show a simple invocation of $\epsilon_X\text{T}_{\text{E}}\text{X}$ without any command line arguments.

```

# extex
This is ExTeX, Version 0.0 (TeX compatibility mode)
**\relax

*\end

No pages of output.
Transcript written on ./texput.log.

```

In this case $\epsilon_X\text{T}_{\text{E}}\text{X}$ enters interaction with the user and asks for an input file. This

2. Getting Started

is indicated by the two asterisks. We have entered `\relax` here to indicate that we are not willing to pass in a file name. The $\epsilon\lambda$ TeX system asks us to enter some command – indicated by the single asterisk. Here we have entered `\end` to indicate that we want to finish the processing. Thus $\epsilon\lambda$ TeX terminates normally.

To be completed.

```
# extex plain
This is ExTeX, Version 0.0 (TeX compatibility mode)
(plain Preloading the plain format: codes, registers, parameters, fonts,
more fonts, macros, math definitions, output routines, hyphenation(hyphen))
*\dump
Beginning to dump on file plain.fmt

*\end

No pages of output.
Transcript written on ./plain.log.
```

2.5.1. Command Line Parameters

The invocation of the executable `extex` can be controlled by large number of command line arguments. Those command line arguments are described in the following list:

<code>

This parameter contains $\epsilon\lambda$ TeX code to be executed directly. The execution is performed after any code specified in an input file. On the command line the code has to start with a backslash. This restriction does not hold for the property settings.

This command line argument sets the property `extex.code`

<file>

This parameter contains the file to read from. A file name may not start with a backslash or an ampercent. It has no default.

This command line argument sets the property `extex.file`.

- *<file>*

This parameter terminates the normal processing of arguments. The next argument – if present – is interpreted as input file. With this construction it is possible to process an input file which starts with one of the special characters `\` or `&`.

This command line argument sets the property `extex.file` if a file argument is present.

-configuration $\langle resource \rangle$

This parameter contains the name of the configuration resource to use. This configuration resource is sought on the class path.

This command line argument sets the property `extex.config`.

-copyright

This command line option produces a copyright notice on the standard output stream and terminates the program afterwards.

& $\langle format \rangle$ **-fmt** $\langle format \rangle$

This parameter contains the name of the format to read. An empty string denotes that no format should be read. This is the default.

This command line argument sets the property `extex.fmt`.

-debug $\langle spec \rangle$

This command line parameter can be used to instruct the program to produce debugging output of several kinds. The debug output is written to the log file. The specification $\langle spec \rangle$ is interpreted left to right. Each character is interpreted according to the following table:

<i>Spec</i>	<i>Description</i>	<i>See</i>
F	This specifier contains the indicator whether or not to trace the searching for input files.	<code>extex.trace.input.files</code>
f	This specifier contains the indicator whether or not to trace the searching for font files.	<code>extex.trace.font.files</code>
M	This specifier contains the indicator whether or not to trace the execution of macros.	<code>extex.trace.macros</code>
T	This specifier contains the indicator whether or not to trace the work of the tokenizer.	<code>extex.trace.tokenizer</code>

The following example shows a possible invocation with this parameter:

```
# extex -debug FfMT abc.tex
This is ExTeX, Version 0.0 (TeX compatibility mode)
...
```

-halt-on-error

This parameter contains the indicator whether the processing should halt after the first error which has been encountered.

This command line argument sets the property `extex.halt.on.error`.

2. Getting Started

-help

This command line option produces a short usage description on the standard output stream and terminates the program afterwards.

-ini

If set to true then act as `iniTeX`. This command line option is defined for compatibility to `TeX` only. In `εTeX` it has no effect at all.

This command line argument sets the property `extex.ini`.

The following example shows a possible invocation with this parameter:

```
# extex -ini abc.tex
This is ExTeX, Version 0.0 (TeX compatibilty mode)
...
```

-interaction *<mode>*

This parameter contains the interaction mode. possible values are the numbers 0...3 and the symbolic names `batchmode` (0), `nonstopmode` (1), `scrollmode` (2), and `errorstopmode` (3).

This command line argument sets the property `extex.interaction`.

The following example shows a possible invocation with this parameter:

```
# extex -interaction batchmode abc.tex
This is ExTeX, Version 0.0 (TeX compatibilty mode)
...
```

-job-name *<name>*

This parameter contains the name of the job. It is overwritten if a file is given to read from. In this case the base name of the input file is used instead.

This command line argument sets the property `extex.jobname`.

-language *<language>*

This parameter contains the name of the locale to be used for the messages.

This command line argument sets the property `extex.lang`.

-output *<format>*

This parameter contains the output format. This logical name is resolved via the configuration.

This command line argument sets the property `extex.output`.

The following example shows a possible invocation with this parameter:

```
# extex -output pdf abc.tex
This is ExTeX, Version 0.0 (TeX compatibilty mode)
```

-progrname *<name>*

This parameter can be used to overrule the name of the program shown in the banner and the version information. The following example shows a possible invocation and the resulting output:

```
# extex -progrname XeTeX -version
This is XeTeX, Version 0.0 (1.4.2_06)
#
```

This command line argument sets the property `extex.progrname`.

-texinputs *<path>*

This parameter contains the additional directories for searching $\epsilon_{\mathcal{X}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ input files. The directories are separated by the system-dependant separator. This separator is a colon (:) on Unix and the semicolon (;) on Windows.

This command line argument sets the property `extex.texinputs`.

-texmfoutputs *<dir>*

This parameter contains the name of the property for the fallback if the output directory fails to be writable.

This command line argument sets the property `extex.outputdir.fallback`.

-texoutputs *<dir>*

This parameter contain the directory where output files should be created.

This command line argument sets the property `extex.outputdir`.

-version

This command line parameter forces that the version information is written to standard output and the program is terminated. The version of $\epsilon_{\mathcal{X}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ is shown and the version of the Java engine in parentheses. The following example shows a possible invocation and the resulting output:

```
# extex -version
This is ExTeX, Version 0.0 (1.4.2_06)
#
```

Command line parameters can be abbreviated up to a unique prefix – and sometimes even more. Thus the following invocations are equivalent:

```
extex -v
extex -ve
extex -ver
extex -vers
extex -versi
extex -versio
extex -version
```

2. Getting Started

2.5.2. Creating Formats

To be completed.

3. Troubleshooting $\epsilon\chi\text{T}\text{E}\text{X}$

This chapter contains some hints in the case of trouble.

3.1. Why are my files not found?

$\epsilon\chi\text{T}\text{E}\text{X}$ has a configurable search for external resources. This search is controlled by several parameters.

To be completed.

3.2. Why are is the log file different from TEX 's?

$\epsilon\chi\text{T}\text{E}\text{X}$ has the goal to produce a visual result comparable to the one of TEX . It has been decided explicitly that the contents of the log file is not considered for compatibility.

The log file is meant for a human reader who should not have any trouble with the differences. The log file is not meant to be a means for communicating with another program.

3. Troubleshooting $\epsilon_X\text{TeX}$

4. The Macro Language of $\epsilon\lambda\text{T}_{\text{E}}\text{X}$

4.1. Primitives of $\epsilon\lambda\text{T}_{\text{E}}\text{X}$

$\epsilon\lambda\text{T}_{\text{E}}\text{X}$ defines a lot of primitives. Those primitives are described below.

The Primitive $\backslash_$

This primitive inserts an explicit space into the current list. This has an effect in horizontal or restricted horizontal modes only. In other modes it has no effect.

The formal description of this primitive is the following:

$\langle \textit{space primitive} \rangle$
 $\rightarrow \backslash_$

Examples:

```
123\ 456
```

```
123\ \ 456
```

The primitive $\backslash_$ is defined in the set `tex`.

The Primitive $\backslash/$

The formal description of this primitive is the following:

$\langle \textit{italic correction} \rangle$
 $\rightarrow \backslash/$

Examples:

```
123\!/456
```

The primitive $\backslash/$ is defined in the set `tex`.

The Primitive `\`

The formal description of this primitive is the following:

$\langle newline \rangle$
 $\rightarrow \backslash$

Examples:

```
\
```

The primitive `\`
is defined in the set `tex`.

The Primitive `\above`

The formal description of this primitive is the following:

$\langle above \rangle$
 $\rightarrow \backslash above$

Examples:

```
{a \above b}
```

The primitive `\above` is defined in the set `tex`.

The Primitive `\abovedisplayshortskip`

`\abovedisplayshortskip` is a skip register. The primitive `\abovedisplayshortskip` is defined in the set `tex`.

The Primitive `\abovedisplayskip`

`\abovedisplayskip` is a skip register. The primitive `\abovedisplayskip` is defined in the set `tex`.

The Primitive `\abovewithdelims`

The formal description of this primitive is the following:

$\langle abovewithdelims \rangle$
 $\rightarrow \backslash abovewithdelims$

Examples:

```
\abovewithdelims
```

The primitive `\abovewithdelims` is defined in the set `tex`.

The Primitive `\accent`

The formal description of this primitive is the following:

$$\langle \textit{accent} \rangle \\ \rightarrow \texttt{\backslash accent} \dots$$

Examples:

```
\accent 13 a
```

The primitive `\accent` is defined in the set `tex`.

The Primitive `\addafterocplist`

`\addafterocplist` is not implemented yet.

The primitive `\addafterocplist` is defined in the set `omega`.

The Primitive `\addbeforeocplist`

`\addbeforeocplist` is not implemented yet.

The primitive `\addbeforeocplist` is defined in the set `omega`.

The Primitive `\adjdemerits`

`\adjdemerits` is a count register. The primitive `\adjdemerits` is defined in the set `tex`.

The Primitive `\advance`

This primitive implements an assignment. The variable given as next tokens is incremented by the quantity given after the optional `by`.

The formal description of this primitive is the following:

$$\begin{aligned} \langle \textit{advance} \rangle & \rightarrow \langle \textit{optional prefix} \rangle \texttt{\backslash advance} \langle \textit{advancable} \rangle \\ \langle \textit{optional prefix} \rangle & \rightarrow \\ & | \quad \texttt{\backslash global} \langle \textit{optional prefix} \rangle \\ \langle \textit{advancable} \rangle & \rightarrow \langle \textit{integer variable} \rangle \langle \textit{optional by} \rangle \langle \textit{number} \rangle \\ & | \quad \langle \textit{dimen variable} \rangle \langle \textit{optional by} \rangle \langle \textit{dimen} \rangle \\ & | \quad \langle \textit{glue variable} \rangle \langle \textit{optional by} \rangle \langle \textit{glue} \rangle \\ & | \quad \langle \textit{muglue variable} \rangle \langle \textit{optional by} \rangle \langle \textit{muglue} \rangle \\ \langle \textit{optional by} \rangle & \end{aligned}$$

4. The Macro Language of $\epsilon\chi\text{TeX}$

\rightarrow [by]
| \langle optional spaces \rangle

Examples:

```
\advance\count12 345
```

```
\advance\count12 by -345
```

The primitive `\advance` is defined in the set `tex`.

The Primitive `\afterassignment`

The primitive `\afterassignment` registers the token to be inserted after the next assignment. Note that there is at most one token to be inserted after the next assignment. Thus the primitive may overwrite any previously registered token.

The formal description of this primitive is the following:

\langle afterassignment \rangle
 \rightarrow `\afterassignment` \langle token \rangle

Examples:

```
\afterassignment\abc
```

```
\afterassignment X
```

```
\afterassignment \~
```

The primitive `\afterassignment` is defined in the set `tex`.

The Primitive `\aftergroup`

This primitive takes the next token and saves it. The saved token will be inserted after the current group has been closed. If several tokens are saved then they will be inserted in the same sequence as they are saved.

The formal description of this primitive is the following:

\langle aftergroup \rangle
 \rightarrow `\aftergroup` \langle token \rangle

Examples:

```
{\aftergroup\~ xyz}
```

```
{\aftergroup\a\aftergroup\b xyz}
```

The primitive `\aftergroup` is defined in the set `tex`.

The Primitive `\atop`

The formal description of this primitive is the following:

$$\langle atop \rangle \rightarrow \backslash atop$$

Examples:

```
\atop
```

The primitive `\atop` is defined in the set `tex`.

The Primitive `\atopwithdelims`

The formal description of this primitive is the following:

$$\langle atopwithdelims \rangle \rightarrow \backslash atopwithdelims$$

Examples:

```
\atopwithdelims
```

The primitive `\atopwithdelims` is defined in the set `tex`.

The Primitive `\badness`

Examples:

```
\count1=\badness
```

The primitive `\badness` is defined in the set `tex`.

The Primitive `\baselineskip`

`\baselineskip` is a skip register. The primitive `\baselineskip` is defined in the set `tex`.

The Primitive `\batchmode`

This primitive is an assignment. The interaction mode is set to batch mode. In batch mode the processing is terminated if the program needs input from the terminal.

The formal description of this primitive is the following:

$$\langle batchmode \rangle \rightarrow \backslash batchmode$$

4. The Macro Language of $\epsilon\chi\text{T}_{\text{E}}\text{X}$

Examples:

```
\batchmode
```

The primitive `\batchmode` is defined in the set `tex`.

The Primitive `\begingroup`

The formal description of this primitive is the following:

$\langle begingroup \rangle$
 $\rightarrow \text{\code\begingroup}$

Examples:

```
\begingroup 123 \endgroup
```

The primitive `\begingroup` is defined in the set `tex`.

The Primitive `\beginL`

`\beginL` is not implemented yet.

The primitive `\beginL` is defined in the set `etex`.

The Primitive `\beginR`

`\beginR` is not implemented yet.

The primitive `\beginR` is defined in the set `etex`.

The Primitive `\belowdisplayshortskip`

`\belowdisplayshortskip` is a skip register. The primitive `\belowdisplayshortskip` is defined in the set `tex`.

The Primitive `\belowdisplayskip`

`\belowdisplayskip` is a skip register. The primitive `\belowdisplayskip` is defined in the set `tex`.

The Primitive `\binoppenalty`

`\binoppenalty` is a count register. The primitive `\binoppenalty` is defined in the set `tex`.

The Primitive `\botmark`

The formal description of this primitive is the following:

`\botmark ...`

Examples:

```
\botmark ...
```

The primitive `\botmark` is defined in the set `tex`.

The Primitive `\botmarks`

`\botmarks` is not implemented yet.

The primitive `\botmarks` is defined in the set `etex`.

The Primitive `\box`

The formal description of this primitive is the following:

$\langle box \rangle$
 $\rightarrow \text{\code{box}} \langle 8\text{-bit number} \rangle$

Examples:

```
\box42
```

The primitive `\box` is defined in the set `tex`.

The Primitive `\boxmaxdepth`

`\boxmaxdepth` is a dimen register. The primitive `\boxmaxdepth` is defined in the set `tex`.

The Primitive `\brokenpenalty`

`\brokenpenalty` is a count register. The primitive `\brokenpenalty` is defined in the set `tex`.

The Primitive `\catcode`

The assignment is controlled by the modifier `\global` and the count parameter `\globaldefs`. Usually the assignment is acting on the current group only. if the integer parameter `\globaldefs` is not 0 or the modifier `\global` is given then the assignment is applied to all groups.

The formal description of this primitive is the following:

4. The Macro Language of $\epsilon_X\text{TeX}$

$\langle catcode \rangle$
 $\rightarrow \backslash catcode \langle 8\text{-bit number} \rangle \langle equals \rangle \langle 4\text{-bit number} \rangle$

Examples:

```
\catcode ...
```

The primitive `\catcode` is defined in the set `tex`.

The Primitive `\char`

The primitive `\char` provides access to any character in the current font. The argument is the numeric value of the character. This value can be any expanded expression resulting in a number of the proper range.

If no proper argument is found then an error is raised.

The formal description of this primitive is the following:

$\langle char \rangle$
 $\rightarrow \backslash char \langle number \rangle$

Examples:

```
\char42  
\char\count1
```

The primitive `\char` is defined in the set `tex`.

The Primitive `\chardef`

The formal description of this primitive is the following:

$\langle chardef \rangle$
 $\rightarrow \backslash chardef \langle control\ sequence \rangle \langle equals \rangle \langle 8\text{-bit number} \rangle$

Examples:

```
\chardef\abc=45
```

```
\chardef\abc 33
```

The primitive `\chardef` is defined in the set `tex`.

The Primitive `\cleaders`

The formal description of this primitive is the following:

$\langle cleaders \rangle$
 $\rightarrow \backslash cleaders \dots$

Examples:

```
\cleaders\hrul\hfill
```

The primitive `\cleaders` is defined in the set `tex`.

The Primitive `\clearocplists`

`\clearocplists` is not implemented yet.

The primitive `\clearocplists` is defined in the set `omega`.

The Primitive `\closein`

The primitive takes one expanded integer argument. This argument denotes a read register which will be closed if it is currently assigned to a file.

The formal description of this primitive is the following:

$\langle closein \rangle$
 $\rightarrow \backslash closein \langle number \rangle$

Examples:

```
\closein5
```

```
\closein\count120
```

The primitive `\closein` is defined in the set `tex`.

The Primitive `\closeout`

The formal description of this primitive is the following:

$\langle closeout \rangle$
 $\rightarrow \backslash closeout \langle number \rangle$

Examples:

```
\closeout5
```

```
\closeout\count120
```

The primitive `\closeout` is defined in the set `tex`.

The Primitive `\clubpenalties`

`\clubpenalties` is not implemented yet.

The primitive `\clubpenalties` is defined in the set `etex`.

The Primitive `\clubpenalty`

`\clubpenalty` is a count register. The primitive `\clubpenalty` is defined in the set `tex`.

The Primitive `\copy`

The formal description of this primitive is the following:

$\langle copy \rangle$
 $\rightarrow \text{\code{\copy}} \langle 8\text{-bit number} \rangle$

Examples:

```
\copy42
```

The primitive `\copy` is defined in the set `tex`.

The Primitive `\count`

The formal description of this primitive is the following:

$\langle count \rangle$
 $\rightarrow \text{\code{\count}} \langle 8\text{-bit number} \rangle \langle equals \rangle \langle number \rangle$

Examples:

```
\count23=-456
```

The primitive `\count` is defined in the set `tex`.

The Primitive `\countdef`

The formal description of this primitive is the following:

$\langle countdef \rangle$
 $\rightarrow \text{\code{\countdef}} \langle control\ sequence \rangle \langle equals \rangle \langle 8\text{-bit number} \rangle$

Examples:

```
\countdef\abc=45
```

```
\countdef\abc 33
```

The primitive `\countdef` is defined in the set `tex`.

The Primitive `\cr`

The formal description of this primitive is the following:

$$\langle cr \rangle \rightarrow \backslash\mathrm{cr}$$

Examples:

```
\cr
```

The primitive `\cr` is defined in the set `tex`.

The Primitive `\crrcr`

The formal description of this primitive is the following:

$$\langle crrcr \rangle \rightarrow \backslash\mathrm{crrcr}$$

Examples:

```
\crrcr
```

The primitive `\crrcr` is defined in the set `tex`.

The Primitive `\csname`

When $\mathrm{T}_E\mathrm{X}$ expands `\csname` it reads to the matching `\endcsname`, expanding tokens as it goes; only character tokens should remain after this expansion has taken place. Then the “expansion” of the entire `\csname...\endcsname` text will be a single control sequence token, defined to be like `\relax` if its meaning is currently undefined.

The formal description of this primitive is the following:

$$\langle csname \rangle \rightarrow \backslash\mathrm{csname} \langle \dots \rangle \backslash\mathrm{endcsname}$$

Examples:

```
\csname abc\endcsname
```

The primitive `\csname` is defined in the set `tex`.

The Primitive `\currentgrouplevel`

...

The formal description of this primitive is the following:

$\langle currentgrouplevel \rangle$
 $\rightarrow \text{\code{\currentgrouplevel}}$

Examples:

```
\the\currentgrouplevel
```

The primitive `\currentgrouplevel` is defined in the set `etex`.

The Primitive `\currentgrouptype`

`\currentgrouptype` is not implemented yet.

The primitive `\currentgrouptype` is defined in the set `etex`.

The Primitive `\currentifbranch`

`\currentifbranch` is not implemented yet.

The primitive `\currentifbranch` is defined in the set `etex`.

The Primitive `\currentiflevel`

`\currentiflevel` is not implemented yet.

The primitive `\currentiflevel` is defined in the set `etex`.

The Primitive `\currentifttype`

`\currentifttype` is not implemented yet.

The primitive `\currentifttype` is defined in the set `etex`.

The Primitive `\day`

`\day` is a count register. The primitive `\day` is defined in the set `tex`.

The Primitive `\deadcycles`

`\deadcycles` is a count register. The primitive `\deadcycles` is defined in the set `tex`.

The Primitive `\def`

The formal description of this primitive is the following:

```

 $\langle def \rangle$ 
   $\rightarrow$   $\langle prefix \rangle \backslash def \langle control\ sequence \rangle \langle parameter\ text \rangle \{ \langle replacement\ text \rangle \}$ 
 $\langle prefix \rangle$ 
   $\rightarrow$ 
  |  $\backslash global \langle prefix \rangle$ 
  |  $\backslash long \langle prefix \rangle$ 
  |  $\backslash outer \langle prefix \rangle$ 

```

Examples:

```
\def#1{--#1--}
```

The primitive `\def` is defined in the set `tex`.

The Primitive `\defaultthyphenchar`

`\defaultthyphenchar` is a count register. The primitive `\defaultthyphenchar` is defined in the set `tex`.

The Primitive `\DefaultInputMode`

`\DefaultInputMode` is not implemented yet.

The primitive `\DefaultInputMode` is defined in the set `omega`.

The Primitive `\DefaultInputTranslation`

`\DefaultInputTranslation` is not implemented yet.

The primitive `\DefaultInputTranslation` is defined in the set `omega`.

The Primitive `\DefaultOutputMode`

`\DefaultOutputMode` is not implemented yet.

The primitive `\DefaultOutputMode` is defined in the set `omega`.

The Primitive `\DefaultOutputTranslation`

`\DefaultOutputTranslation` is not implemented yet.

The primitive `\DefaultOutputTranslation` is defined in the set `omega`.

The Primitive `\defaultskewchar`

`\defaultskewchar` is a count register. The primitive `\defaultskewchar` is defined in the set `tex`.

The Primitive `\delcode`

The $\text{T}_{\text{E}}\text{X}$ encoding interprets the number as 27 bit hex number: "csyylxx. Here the digits have the following meaning:

c the math class of this delimiter. It has a range from 0 to 7.

l the family for the large character. It has a range from 0 to 15.

xx the character code of the large character.

s the family for the small character. It has a range from 0 to 15.

yy the character code of the small character.

The formal description of this primitive is the following:

$\langle delcode \rangle$
 $\rightarrow \backslash delcode \langle 8\text{-bit number} \rangle \langle equals \rangle \langle 8\text{-bit number} \rangle$

Examples:

```
\delcode'x="123456
```

The primitive `\delcode` is defined in the set `tex`.

The Primitive `\delimiter`

The formal description of this primitive is the following:

$\langle delimiter \rangle$
 $\rightarrow \backslash delimiter$

Examples:

```
\delimiter "426830A
```

The primitive `\delimiter` is defined in the set `tex`.

The Primitive `\delimiterfactor`

`\delimiterfactor` is a count register. The primitive `\delimiterfactor` is defined in the set `tex`.

The Primitive `\delimitershortfall`

`\delimitershortfall` is a `dimen` register. The primitive `\delimitershortfall` is defined in the set `tex`.

The Primitive `\detokenize`

`\detokenize` is not implemented yet.

The primitive `\detokenize` is defined in the set `etex`.

The Primitive `\dimen`

The primitive `\dimen` is defined in the set `tex`.

The Primitive `\dimendef`

The formal description of this primitive is the following:

$\langle \textit{dimendef} \rangle$
 $\rightarrow \texttt{\backslash dimendef} \langle \textit{control sequence} \rangle \langle \textit{equals} \rangle \langle \textit{8-bit number} \rangle$

Examples:

```
\dimendef\abc=45
```

```
\dimendef\abc 33
```

The primitive `\dimendef` is defined in the set `tex`.

The Primitive `\dimenexpr`

`\dimenexpr` is not implemented yet.

The primitive `\dimenexpr` is defined in the set `etex`.

The Primitive `\discretionary`

The formal description of this primitive is the following:

$\langle \textit{discretionary} \rangle$
 $\rightarrow \texttt{\backslash discretionary} \dots$

Examples:

```
\discretionary{f-}{fi}{ffi}  
\discretionary{-}{}{}
```

The primitive `\discretionary` is defined in the set `tex`.

The Primitive `\displayindent`

`\displayindent` is a dimen register. The primitive `\displayindent` is defined in the set `tex`.

The Primitive `\displaylimits`

The formal description of this primitive is the following:

$\langle displaylimits \rangle$
 $\rightarrow \text{\code{\displaylimits}}$

Examples:

```
\displaylimits
```

The primitive `\displaylimits` is defined in the set `tex`.

The Primitive `\displaystyle`

The formal description of this primitive is the following:

$\langle displaystyle \rangle$
 $\rightarrow \text{\code{\displaystyle}}$

Examples:

```
\displaystyle
```

The primitive `\displaystyle` is defined in the set `tex`.

The Primitive `\displaywidowpenalties`

`\displaywidowpenalties` is not implemented yet.

The primitive `\displaywidowpenalties` is defined in the set `etex`.

The Primitive `\displaywidowpenalty`

`\displaywidowpenalty` is a count register. The primitive `\displaywidowpenalty` is defined in the set `tex`.

The Primitive `\displaywidth`

`\displaywidth` is a dimen register. The primitive `\displaywidth` is defined in the set `tex`.

The Primitive `\divide`

This primitive implements an assignment. The variable given as next tokens is divided by the quantity given after the optional `by`.

The formal description of this primitive is the following:

```

<divide>
  → \divide <dividable>
<dividable>
  → <integer variable> <optional by> <8-bit number>
  | <dimen variable> <optional by> <8-bit number>
  | <glue variable> <optional by> <8-bit number>
  | <muglue variable> <optional by> <8-bit number>
<optional by>
  → [by]
  | <optional spaces>

```

Examples:

```
\divide\count12 345
```

```
\divide\count12 by -345
```

The primitive `\divide` is defined in the set `tex`.

The Primitive `\doublehyphendemerits`

`\doublehyphendemerits` is a count register. The primitive `\doublehyphendemerits` is defined in the set `tex`.

The Primitive `\dp`

The primitive `\dp` refers to the depth of a box register. It can be used in various contexts.

Execution of the Primitive

If the primitive is used in a context it initiated an assignment to the actual depth of the box register. This has an effect only in the case that the box register is not void.

The formal description of this primitive is the following:

```

<dp>
  → <optional prefix> \dp <8-bit number> <equals> <dimen>
<optional prefix>
  →
  | \global <optional prefix>

```

4. The Macro Language of $\epsilon\chi\TeX$

Examples:

```
\dp42 = 12mm
```

```
\dp42 = \dimen3
```

Expansion of the Primitive

In an expansion context the primitive results in the the current depth of the given box register. In case that the box register is empty the result is 0 pt.

The formal description of this primitive is the following:

$\backslash\mathrm{dp}$ $\langle 8\text{-bit number} \rangle$

Examples:

```
\dimen0 = \dp42
```

Conversion to a Count

Interaction with $\backslash\mathrm{the}$

The primitive $\backslash\mathrm{dp}$ is defined in the set `tex`.

The Primitive $\backslash\mathrm{dump}$

The primitive writes out the current state of the interpreter to an format file. This format file can be read back in to restore the saved state.

The primitive can be used outside of any group only.

The formal description of this primitive is the following:

$\langle dump \rangle$
 $\rightarrow \backslash\mathrm{dump}$

Examples:

```
\dump
```

The primitive $\backslash\mathrm{dump}$ is defined in the set `tex`.

The Primitive `\edef`

The formal description of this primitive is the following:

$$\begin{aligned} \langle \textit{edef} \rangle & \rightarrow \langle \textit{prefix} \rangle \backslash \textit{edef} \langle \textit{control sequence} \rangle \langle \textit{parameter text} \rangle \{ \langle \textit{replacement text} \rangle \} \\ \langle \textit{prefix} \rangle & \rightarrow \\ & \begin{array}{|l} \backslash \textit{global} \langle \textit{prefix} \rangle \\ \backslash \textit{long} \langle \textit{prefix} \rangle \\ \backslash \textit{outer} \langle \textit{prefix} \rangle \end{array} \end{aligned}$$

Examples:

```
\edef#1{--#1--}
```

The primitive `\edef` is defined in the set `tex`.

The Primitive `\else`

The formal description of this primitive is the following:

$$\begin{aligned} \langle \textit{else} \rangle & \rightarrow \backslash \textit{else} \langle \dots \rangle \end{aligned}$$

Examples:

```
\ifnum 1<2\else no\fi
```

The primitive `\else` is defined in the set `tex`.

The Primitive `\emergencystretch`

`\emergencystretch` is a dimen register. The primitive `\emergencystretch` is defined in the set `tex`.

The Primitive `\end`

The formal description of this primitive is the following:

$$\begin{aligned} \langle \textit{end} \rangle & \rightarrow \backslash \textit{end} \end{aligned}$$

Examples:

```
\end
```

The primitive `\end` is defined in the set `tex`.

The Primitive `\endcsname`

The macro `\endcsname` is used in combination with the macro `\csname` only. Whenever a `\endcsname` is seen alone it must be an error. Thus this primitive produces an error message in any case.

The formal description of this primitive is the following:

$\langle\textit{endcsname}\rangle$
→ `\endcsname`

Examples:

```
\csname abc\endcsname
```

The primitive `\endcsname` is defined in the set `tex`.

The Primitive `\endgroup`

The formal description of this primitive is the following:

$\langle\textit{endgroup}\rangle$
→ `\endgroup`

Examples:

```
\begingroup 123 \endgroup
```

The primitive `\endgroup` is defined in the set `tex`.

The Primitive `\endinput`

The primitive `\endinput` closes the topmost file input stream. All tokens collected for this input stream and the ones above are discarded. This means that you can place arbitrary text behind this primitive in a file. This text is ignored immediately.

The formal description of this primitive is the following:

$\langle\textit{endinput}\rangle$
→ `\endinput`

Examples:

```
\endinput
```

The primitive `\endinput` is defined in the set `tex`.

The Primitive `\endL`

`\endL` is not implemented yet.

The primitive `\endL` is defined in the set `etex`.

The Primitive `\endlinechar`

`\endlinechar` is a count register. The primitive `\endlinechar` is defined in the set `tex`.

The Primitive `\endR`

`\endR` is not implemented yet.

The primitive `\endR` is defined in the set `etex`.

The Primitive `\eqno`

The formal description of this primitive is the following:

$$\langle eqno \rangle \rightarrow \backslash eqno$$

Examples:

```
\eqno
```

The primitive `\eqno` is defined in the set `tex`.

The Primitive `\errhelp`

`\errhelp` is a toks register. The primitive `\errhelp` is defined in the set `tex`.

The Primitive `\errmessage`

The primitive `\errmessage` takes one argument. This argument is an expanded list of tokens. Those tokens are presented as error message

The formal description of this primitive is the following:

$$\langle eqno \rangle \rightarrow \backslash errmessage \langle tokens \rangle$$

Examples:

```
\errmessage{}
```

The primitive `\errmessage` is defined in the set `tex`.

The Primitive `\errorcontextlines`

`\errorcontextlines` is a count register. The primitive `\errorcontextlines` is defined in the set `tex`.

The Primitive `\errorstopmode`

The formal description of this primitive is the following:

$\langle errorstopmode \rangle$
 $\rightarrow \text{\errorstopmode}$

Examples:

```
\errorstopmode
```

The primitive `\errorstopmode` is defined in the set `tex`.

The Primitive `\escapechar`

`\escapechar` is a count register. The primitive `\escapechar` is defined in the set `tex`.

The Primitive `\eTeXrevision`

`\eTeXrevision` is a toks register. The primitive `\eTeXrevision` is defined in the set `etex`.

The Primitive `\eTeXversion`

`\eTeXversion` is a count register. The primitive `\eTeXversion` is defined in the set `etex`.

The Primitive `\everycr`

`\everycr` is a toks register. The primitive `\everycr` is defined in the set `tex`.

The Primitive `\everydisplay`

`\everydisplay` is a toks register. The primitive `\everydisplay` is defined in the set `tex`.

The Primitive `\everyeof`

`\everyeof` is a toks register. The primitive `\everyeof` is defined in the set `etex`.

The Primitive `\everyhbox`

`\everyhbox` is a toks register. The primitive `\everyhbox` is defined in the set `tex`.

The Primitive `\everyjob`

`\everyjob` is a toks register. The primitive `\everyjob` is defined in the set `tex`.

The Primitive `\everymath`

`\everymath` is a toks register. The primitive `\everymath` is defined in the set `tex`.

The Primitive `\everypar`

`\everypar` is a toks register. The primitive `\everypar` is defined in the set `tex`.

The Primitive `\everyvbox`

`\everyvbox` is a toks register. The primitive `\everyvbox` is defined in the set `tex`.

The Primitive `\exhyphenpenalty`

`\exhyphenpenalty` is a count register. The primitive `\exhyphenpenalty` is defined in the set `tex`.

The Primitive `\expandafter`

$\mathrm{T}_E\mathrm{X}$ first reads the token that comes immediately after `\expandafter`, without expanding it; let's call this token t . Then $\mathrm{T}_E\mathrm{X}$ reads the token that comes after t (and possibly more tokens, if that token has an argument), replacing it by its expansion. Finally $\mathrm{T}_E\mathrm{X}$ puts t back in front of that expansion.

The formal description of this primitive is the following:

$\langle \textit{expandafter} \rangle$
 $\rightarrow \texttt{\backslash expandafter} \langle \textit{control sequence} \rangle \dots$

Examples:

```
\expandafter ...
```

The primitive `\expandafter` is defined in the set `tex`.

The Primitive `\export`

The primitive `\export` takes a list of tokens and saves them away for an associated `\import`. The tokens in the list are either control sequence tokens or active characters. All other tokens are ignored.

The formal description of this primitive is the following:

$$\langle export \rangle \rightarrow \texttt{\backslash export} \langle replacement\ text \rangle$$

Examples:

```
\export{\a\b}
```

The primitive `\export` is defined in the set `namespace`.

The Primitive `\fam`

`\fam` is a count register. The primitive `\fam` is defined in the set `tex`.

The Primitive `\fi`

This primitive indicates the end of an conditional. As such it can not appear alone but only in combination with a preceeding `\if*`.

The formal description of this primitive is the following:

$$\langle fi \rangle \rightarrow \texttt{\backslash fi}$$

Examples:

```
\fi
```

The primitive `\fi` is defined in the set `tex`.

The Primitive `\finalhyphendemerits`

`\finalhyphendemerits` is a count register. The primitive `\finalhyphendemerits` is defined in the set `tex`.

The Primitive `\firstmark`

The formal description of this primitive is the following:

```
\firstmark ...
```

Examples:

```
\firstmark ...
```

The primitive `\firstmark` is defined in the set `tex`.

The Primitive `\firstmarks`

`\firstmarks` is not implemented yet.

The primitive `\firstmarks` is defined in the set `etex`.

The Primitive `\floatingpenalty`

`\floatingpenalty` is a count register. The primitive `\floatingpenalty` is defined in the set `tex`.

The Primitive `\font`

The primitive `\font` can be used to load a font with some specified properties and assign it to a control sequence. The primary option is the specification of a size for the font. If no size is given then the font is loaded at its design size.

An exact size can be specified with the `at` keyword. The dimension following this keyword determines the size of the font.

The design size can be multiplied by a scale factor. This scale factor is given as number after the keyword `scaled`. The value given is 1000 times the scale factor to be used.

This primitive is an assignment.

The formal description of this primitive is the following:

```

<font>
  → \font <control sequence> <equals> <font name> <options>
<options>
  → <option>
  | <option> <options>
<option>
  → [scaled] <number>
  | [at] <size...>
  | [noligatures]
  | [nokerning]
  | [letterspaced]
```

Examples

In the following example the font `cmr12` is loaded at its design size. The macro `\myfont` is bound to this font.

4. The Macro Language of $\epsilon\chi\TeX$

```
\font\myfont=cmr12
```

In the following example the font cmr12 is loaded at the size 15pt. The macro `\myfont` is bound to this font.

```
\font\myfont=cmr12 at 15pt
```

In the following example the font cmr12 is loaded at the double design size. The scale factor 2000 is divided by 1000 to get the effective scaling factor. The macro `\myfont` is bound to this font.

```
\font\magnifiedfivem=cmr5 scaled 2000
```

In the following example the font cmr10 is loaded at the size of 12 true pt. The macro `\myfont` is bound to this font.

```
\font\second=cmr10 at 12truept
```

The primitive `\font` is defined in the set `tex`.

The Primitive `\fontchar dp`

The formal description of this primitive is the following:

`\fontchar dp`

Examples:

```
\fontchar dp\tenrm ‘a
```

The primitive `\fontchar dp` is defined in the set `etex`.

The Primitive `\fontchar ht`

The formal description of this primitive is the following:

`\fontchar ht`

Examples:

```
\fontchar ht\tenrm ‘a
```

The primitive `\fontchar ht` is defined in the set `etex`.

The Primitive `\fontcharic`

The formal description of this primitive is the following:

`\fontcharic`

Examples:

```
\fontcharic\tenrm ‘a
```

The primitive `\fontcharic` is defined in the set `etex`.

The Primitive `\fontcharwd`

The formal description of this primitive is the following:

`\fontcharwd`

Examples:

```
\fontcharwd\tenrm ‘a
```

The primitive `\fontcharwd` is defined in the set `etex`.

The Primitive `\fontdimen`

The primitive `\fontdimen` can be used to set a font dimension value. Each font has an arbitrary number of `dimen` values which are addressed by an numerical index in $\mathrm{T}_E\mathrm{X}$. In $\epsilon_X\mathrm{T}_E\mathrm{X}$ this has been extended to arbitrary strings.

The primitive expands to the value in a right hand context.

The formal description of this primitive is the following:

`\fontdimen` *<8-bit number>* *<equals>* *<dimen>*

Examples:

```
\fontdimen13\ff=5pt
```

```
\the\fontdimen13\ff
```

```
\the\fontdimen{em}\ff=8pt
```

The primitive `\fontdimen` is defined in the set `tex`.

The Primitive `\fontname`

The primitive `\fontname` can be used to retrieve the name of a font. It takes a font specification as argument. It expands to the name of the font. If this font is not loaded at its design size then the actual size is appended after the tokens `at`. All tokens produced this way are *other* tokens except of the spaces. This means that even the letters are of category *other*.

The primitive `\fontname` is defined in the set `tex`.

The Primitive `\futurelet`

The formal description of this primitive is the following:

$\langle futurelet \rangle$
 $\rightarrow \text{\texttt{\textbackslash futurelet}} \langle control\ sequence \rangle \langle token \rangle \dots$

Examples:

```
\futurelet ...
```

The primitive `\futurelet` is defined in the set `tex`.

The Primitive `\gdef`

The formal description of this primitive is the following:

$\langle gdef \rangle$
 $\rightarrow \langle prefix \rangle \text{\texttt{\textbackslash gdef}} \langle control\ sequence \rangle \langle parameter\ text \rangle \{ \langle replacement\ text \rangle \}$
 $\langle prefix \rangle$
 \rightarrow
 $\begin{array}{l} | \quad \text{\texttt{\textbackslash global}} \langle prefix \rangle \\ | \quad \text{\texttt{\textbackslash long}} \langle prefix \rangle \\ | \quad \text{\texttt{\textbackslash outer}} \langle prefix \rangle \end{array}$

Examples:

```
\gdef#1{--#1--}
```

The primitive `\gdef` is defined in the set `tex`.

The Primitive `\global`

The primitive `\global` is a prefix macro. It does not do anything by its own but works in combination with a following primitive token only. If the following token constitutes an assignment then the assignment is not restricted to the current group but acts globally in all groups.

If the following command token does not happen to be an operation for which the global modifier is applicable then a warning might be raised.

The formal description of this primitive is the following:

$\langle global \rangle$
 $\rightarrow \backslash global \langle \dots \rangle$

Examples

The following example shows that two macros defined in a group. The first macro falls back to its previous binding when the group is closed. The second macro has the same binding in all groups. defined.

```
\begingroup
  \def\{a\}{123}
  \global\def\{b\}{123}
\endgroup
```

The following example shows that two count registers are set in a group. The first count register keeps its value until the group is closed and falls back to the value it had when the group has been entered. The second count register keeps its value even when the group is closed.

```
\begingroup
  \count1=123
  \global\count2=45
\endgroup
```

The primitive `\global` is defined in the set `tex`.

The Primitive `\globaldefs`

`\globaldefs` is a count register. The primitive `\globaldefs` is defined in the set `tex`.

The Primitive `\glueexpr`

`\glueexpr` is not implemented yet.

The primitive `\glueexpr` is defined in the set `etex`.

The Primitive `\glueshrink`

`\glueshrink` is not implemented yet.

The primitive `\glueshrink` is defined in the set `etex`.

The Primitive `\glueshrinkorder`

`\glueshrinkorder` is not implemented yet.

The primitive `\glueshrinkorder` is defined in the set `etex`.

The Primitive `\gluestretch`

`\gluestretch` is not implemented yet.

The primitive `\gluestretch` is defined in the set `etex`.

The Primitive `\gluestretchorder`

`\gluestretchorder` is not implemented yet.

The primitive `\gluestretchorder` is defined in the set `etex`.

The Primitive `\halign`

The formal description of this primitive is the following:

```
 $\langle halign \rangle$   
   $\rightarrow$  \halign  $\langle box\ specification \rangle$  {  $\langle preamble \rangle$  \cr  $\langle rows \rangle$  }  
 $\langle box\ specification \rangle$   
   $\rightarrow$   
  | to  $\langle rule\ dimension \rangle$   
  | spread  $\langle rule\ dimension \rangle$   
 $\langle rows \rangle$   
   $\rightarrow$   
  |  $\langle row \rangle$   $\langle rows \rangle$   
 $\langle preamble \rangle$   
   $\rightarrow$  ...
```

Examples:

```
\halign
```

The primitive `\halign` is defined in the set `tex`.

The Primitive `\hangafter`

`\hangafter` is a count register. The primitive `\hangafter` is defined in the set `tex`.

The Primitive `\hangindent`

`\hangindent` is a dimen register. The primitive `\hangindent` is defined in the set `tex`.

The Primitive `\hbadness`

`\hbadness` is a count register. The primitive `\hbadness` is defined in the set `tex`.

The Primitive `\hbox`

The contents of the toks register `\everyhbox` is inserted at the beginning of the horizontal material of the box.

The formal description of this primitive is the following:

$$\begin{aligned}
 \langle hbox \rangle &\rightarrow \backslash hbox \langle box\ specification \rangle \{ \langle horizontal\ material \rangle \} \\
 \langle box\ specification \rangle &\rightarrow \\
 &\quad | \quad to \langle rule\ dimension \rangle \\
 &\quad | \quad spread \langle rule\ dimension \rangle
 \end{aligned}$$

Examples:

```
\hbox{abc}
```

```
\hbox to 120pt{abc}
```

```
\hbox spread 12pt{abc}
```

The tokens parameter is used in `/hbox`. The tokens contained are inserted at the beginning of the horizontal material of the `hbox`.

The primitive `\hbox` is defined in the set `tex`.

The Primitive `\hfi`

The formal description of this primitive is the following:

$$\begin{aligned}
 \langle hfi \rangle &\rightarrow \backslash hfi
 \end{aligned}$$

Examples:

```
\hfi
```

The primitive `\hfi` is defined in the set `omega`.

The Primitive `\hfil`

The formal description of this primitive is the following:

$$\langle hfil \rangle \rightarrow \backslash hfil$$

Examples:

```
\hfil
```

The primitive `\hfil` is defined in the set `tex`.

The Primitive `\hfill`

The formal description of this primitive is the following:

$$\langle hfill \rangle \rightarrow \backslash hfill$$

Examples:

```
\hfill
```

The primitive `\hfill` is defined in the set `tex`.

The Primitive `\hfilneg`

The formal description of this primitive is the following:

$$\langle hfilneg \rangle \rightarrow \backslash hfilneg$$

Examples:

```
\hfilneg
```

The primitive `\hfilneg` is defined in the set `tex`.

The Primitive `\hfuzz`

`\hfuzz` is a dimen register. The primitive `\hfuzz` is defined in the set `tex`.

The Primitive `\hoffset`

`\hoffset` is a dimen register. The primitive `\hoffset` is defined in the set `tex`.

The Primitive `\holdinginserts`

`\holdinginserts` is a count register. The primitive `\holdinginserts` is defined in the set `tex`.

The Primitive `\hrule`

This primitive produces a horizontal rule. This is a rectangular area of specified dimensions. If not overwritten the width and depth are 0pt and the height is 0.4 pt (26214 sp).

The formal description of this primitive is the following:

```

⟨hrule⟩
  → \hrule ⟨rule specification⟩
⟨rule specification⟩
  → ⟨optional spaces⟩
  |   ⟨rule dimension⟩ ⟨rule specification⟩
⟨rule dimension⟩
  → width ⟨dimen⟩
  |   height ⟨dimen⟩
  |   depth ⟨dimen⟩

```

The color from the typographic context is taken as foreground color for the rule. The default color is black.

Examples:

```
\hrule
```

```
\hrule width 2pt
```

```
\hrule width 2pt depth 3mm height \dimen4
```

The primitive `\hrule` is defined in the set `tex`.

The Primitive `\hsize`

`\hsize` is a dimen register. The primitive `\hsize` is defined in the set `tex`.

The Primitive `\hskip`

The formal description of this primitive is the following:

```

⟨hskip⟩
  → \hskip ⟨Glue⟩

```

4. The Macro Language of $\epsilon_X\text{TeX}$

Examples:

```
\hskip 1em plus 1pt minus 1pt
```

The primitive `\hskip` is defined in the set `tex`.

The Primitive `\hss`

The formal description of this primitive is the following:

$$\langle hss \rangle \rightarrow \text{\hss}$$

Examples:

```
\hss
```

The primitive `\hss` is defined in the set `tex`.

The Primitive `\ht`

The formal description of this primitive is the following:

$$\langle ht \rangle \rightarrow \text{\ht} \langle 8\text{-bit number} \rangle \langle equals \rangle \langle dimen \rangle$$

Examples:

```
\ht42
```

The primitive `\ht` is defined in the set `tex`.

The Primitive `\hyphenation`

The primitive `\hyphenation` is defined in the set `tex`.

The Primitive `\hyphenchar`

The formal description of this primitive is the following:

$$\text{\hyphenchar} \langle font \rangle \langle equals \rangle \langle 8\text{-bit number} \rangle$$

Examples:

```
\hyphenchar\font=132
```

Incompatibility

The TeXbook gives no indication how the primitive should react for negative values – except -1. The implementation of T_EX allows to store and retrieve arbitrary negative values. This behaviour of T_EX is not preserved in $\varepsilon_X\text{T}_E\text{X}$.

The primitive `\hyphenchar` is defined in the set `tex`.

The Primitive `\hyphenpenalty`

`\hyphenpenalty` is a count register. The primitive `\hyphenpenalty` is defined in the set `tex`.

The Primitive `\if`

The primitive expands the tokens following it until two unexpandable tokens are found. The conditional is true iff the character codes of the two tokens agree.

The formal description of this primitive is the following:

$$\begin{array}{l} \langle if \rangle \\ \rightarrow \quad \backslash\text{if } \langle token_1 \rangle \langle token_2 \rangle \langle true\ text \rangle \backslash\text{fi} \\ | \quad \backslash\text{if } \langle token_1 \rangle \langle token_2 \rangle \langle true\ text \rangle \backslash\text{else } \langle false\ text \rangle \backslash\text{fi} \end{array}$$

Examples:

```
\if\aa\aa ok \fi
```

The primitive `\if` is defined in the set `tex`.

The Primitive `\ifcase`

$$\begin{array}{l} \langle ifcase \rangle \\ \rightarrow \quad \backslash\text{ifcase } \dots \end{array}$$

The primitive `\ifcase` is defined in the set `tex`.

The Primitive `\ifcat`

$$\begin{array}{l} \langle ifcat \rangle \\ \rightarrow \quad \backslash\text{ifcat } \dots \end{array}$$

The primitive `\ifcat` is defined in the set `tex`.

The Primitive `\ifcsname`

`\ifcsname` is not implemented yet.

The primitive `\ifcsname` is defined in the set `etex`.

The Primitive `\ifdefined`

Copied of the eTeX reference.

similar in effect to `\unless \ifx \undefined`, but does not require `\undefined` to actually be undefined, since no explicit comparison is made with any particular control sequence.

The formal description of this primitive is the following:

Examples:

```
\ifdefined\TESTNAME\else not\fi defined
```

The primitive `\ifdefined` is defined in the set `etex`.

The Primitive `\ifdim`

The formal description of this primitive is the following:

```
<ifdim>
  → \ifdim <dimen> <op> <dimen> <true text> \fi
  | \ifdim <dimen> <op> <dimen> <true text> \else <>false text> \fi
<op>
  → [<]
  | [=]
  | [>]
```

The primitive `\ifdim` is defined in the set `tex`.

The Primitive `\ifeof`

This primitive tests for end of file on the given read register. The read register is specified as a (expanded) number.

The formal description of this primitive is the following:

```
<ifeof>
  → \ifeof <number> <true text> \fi
  | \ifeof <number> <true text> \else <>false text> \fi
```

Examples:

```
\ifeof 3 -E-O-F- \else ready \fi
```

The primitive `\ifeof` is defined in the set `tex`.

The Primitive `\iffalse`

The primitive does not take any further arguments. The conditional is always false. Thus only the else branch is expanded.

The formal description of this primitive is the following:

```

<iffalse>
  → \iffalse <true text> \fi
  |   \iffalse <true text> \else <false text> \fi

```

Examples:

```
\iffalse abc \fi
```

The primitive `\iffalse` is defined in the set `tex`.

The Primitive `\ifhbox`

The primitive takes one expanded integer argument. The conditional is true iff the box denoted by the argument is a horizontal box.

The formal description of this primitive is the following:

```

<ifhbox>
  → \ifhbox <number> <true text> \fi
  |   \ifhbox <number> <true text> \else <false text> \fi

```

Examples:

```
\ifhbox255 abc \fi
```

```
\ifhbox\count120 abc \fi
```

The primitive `\ifhbox` is defined in the set `tex`.

The Primitive `\ifhmode`

The primitive does not take any further arguments. The conditional is true iff the typesetter is in a horizontal mode. This is either the restricted horizontal vertical mode or the horizontal mode.

The formal description of this primitive is the following:

```

<ifhmode>
  → \ifhmode <true text> \fi
  |   \ifhmode <true text> \else <false text> \fi

```

4. The Macro Language of $\epsilon\chi\TeX$

Examples:

```
\ifhmode abc \fi
```

The primitive `\ifhmode` is defined in the set `tex`.

The Primitive `\ifinner`

The primitive does not take any further arguments. The conditional is true iff the typesetter is in an internal mode. This is either the internal vertical mode, the restricted horizontal mode, or the math mode (non-display).

The formal description of this primitive is the following:

```
 $\langle ifinner \rangle$   
→ \ifinner  $\langle true\ text \rangle$  \fi  
| \ifinner  $\langle true\ text \rangle$  \else  $\langle false\ text \rangle$  \fi
```

Examples:

```
\ifinner abc \fi
```

The primitive `\ifinner` is defined in the set `tex`.

The Primitive `\ifmmode`

The primitive does not take any further arguments. The conditional is true iff the typesetter is in math mode or display math mode.

The formal description of this primitive is the following:

```
 $\langle ifmmode \rangle$   
→ \ifmmode  $\langle true\ text \rangle$  \fi  
| \ifmmode  $\langle true\ text \rangle$  \else  $\langle false\ text \rangle$  \fi
```

Examples:

```
\ifmmode abc \fi
```

The primitive `\ifmmode` is defined in the set `tex`.

The Primitive `\ifnum`

The formal description of this primitive is the following:

```
 $\langle ifnum \rangle$   
→ \ifnum  $\langle number \rangle$   $\langle op \rangle$   $\langle number \rangle$   $\langle true\ text \rangle$  \fi  
| \ifodd  $\langle number \rangle$   $\langle op \rangle$   $\langle number \rangle$   $\langle true\ text \rangle$  \else  $\langle false\ text \rangle$  \fi
```


$\langle op \rangle$
 \rightarrow [$<$]
 $|$ [$=$]
 $|$ [$>$]

Examples:

```
\ifodd\count0 abc \fi
```

The primitive `\ifnum` is defined in the set `tex`.

The Primitive `\ifodd`

The primitive takes one expanded integer argument. The conditional is true iff the argument is odd.

The formal description of this primitive is the following:

$\langle ifodd \rangle$
 \rightarrow `\ifodd $\langle number \rangle$ $\langle true\ text \rangle$ \fi`
 $|$ `\ifodd $\langle number \rangle$ $\langle true\ text \rangle$ \else $\langle false\ text \rangle$ \fi`

Examples:

```
\ifodd\count0 abc \fi
```

The primitive `\ifodd` is defined in the set `tex`.

The Primitive `\iftrue`

The primitive does not take any further arguments. The conditional is always true. Thus only the then branch is expanded.

The formal description of this primitive is the following:

$\langle iftrue \rangle$
 \rightarrow `\iftrue $\langle true\ text \rangle$ \fi`
 $|$ `\ifture $\langle true\ text \rangle$ \else $\langle false\ text \rangle$ \fi`

Examples:

```
\iftrue abc \fi
```

The primitive `\iftrue` is defined in the set `tex`.

The Primitive `\ifvbox`

The primitive takes one expanded integer argument. The conditional is true iff the box denoted by the argument is a vertical box.

The formal description of this primitive is the following:

$\langle ifvbox \rangle$
→ `\ifvbox $\langle number \rangle$ $\langle true text \rangle$ \fi`
| `\ifvbox $\langle number \rangle$ $\langle true text \rangle$ \else $\langle false text \rangle$ \fi`

Examples:

```
\ifvbox255 abc \fi
```

```
\ifvbox\count120 abc \fi
```

The primitive `\ifvbox` is defined in the set `tex`.

The Primitive `\ifvmode`

The primitive does not take any further arguments. The conditional is true iff the typesetter is in a vertical mode. This is either the internal vertical mode or the vertical mode.

The formal description of this primitive is the following:

$\langle ifvmode \rangle$
→ `\ifvmode $\langle true text \rangle$ \fi`
| `\ifvmode $\langle true text \rangle$ \else $\langle false text \rangle$ \fi`

Examples:

```
\ifvmode abc \fi
```

The primitive `\ifvmode` is defined in the set `tex`.

The Primitive `\ifvoid`

The primitive takes one expanded integer argument. The conditional is true iff the box denoted by the argument is void.

The formal description of this primitive is the following:

$\langle ifvoid \rangle$
→ `\ifvoid $\langle number \rangle$ $\langle true text \rangle$ \fi`
| `\ifvoid $\langle number \rangle$ $\langle true text \rangle$ \else $\langle false text \rangle$ \fi`

Examples:

```
\ifvoid255 abc \fi
```

```
\ifvoid\count120 abc \fi
```

The primitive `\ifvoid` is defined in the set `tex`.

The Primitive `\ifx`

The formal description of this primitive is the following:

```
 $\langle ifx \rangle$ 
   $\rightarrow$  \ifx  $\langle token_1 \rangle$   $\langle token_2 \rangle$ ;  $\langle true\ text \rangle$  \fi
  | \ifx  $\langle token_1 \rangle$   $\langle token_2 \rangle$   $\langle true\ text \rangle$  \else  $\langle false\ text \rangle$  \fi
```

Examples:

```
\ifx\ a\ x ok \fi
```

The primitive `\ifx` is defined in the set `tex`.

The Primitive `\ignorespaces`

The formal description of this primitive is the following:

```
 $\langle ignorespaces \rangle$ 
   $\rightarrow$  \ignorespaces
```

Examples:

```
\ignorespaces
```

The primitive `\ignorespaces` is defined in the set `tex`.

The Primitive `\immediate`

The formal description of this primitive is the following:

```
 $\langle immediate \rangle$ 
   $\rightarrow$  \immediate ...
```

Examples:

```
\immediate\write1{abc}
```

The primitive `\immediate` is defined in the set `tex`.

The Primitive `\import`

The formal description of this primitive is the following:

$\langle import \rangle$
 $\rightarrow \text{\texttt{\textbackslash import}} \langle replacement\ text \rangle$

Examples:

```
\import{de.dante.dtk}
```

The primitive `\import` is defined in the set `namespace`.

The Primitive `\indent`

The formal description of this primitive is the following:

$\langle indent \rangle$
 $\rightarrow \text{\texttt{\textbackslash indent}}$

Examples:

The primitive `\indent` is defined in the set `tex`.

The Primitive `\input`

The formal description of this primitive is the following:

$\langle input \rangle$
 $\rightarrow \text{\texttt{\textbackslash input}} \langle filename \rangle$

Examples: The traditional version of the file name parsing allows the following syntax:

```
\input file.name
```

If the parsing is not configured to be strict then the following syntax is allowed as well:

```
\input{file.name}
```

The primitive `\input` is defined in the set `tex`.

The Primitive `\inputlineno`

Examples:

```
\count1=\inputlineno
```

The primitive `\inputlineno` is defined in the set `tex`.

The Primitive `\InputMode`

`\InputMode` is not implemented yet.

The primitive `\InputMode` is defined in the set `omega`.

The Primitive `\InputTranslation`

`\InputTranslation` is not implemented yet.

The primitive `\InputTranslation` is defined in the set `omega`.

The Primitive `\insert`

The formal description of this primitive is the following:

$$\langle \textit{insert} \rangle \\ \rightarrow \text{\texttt{\backslash insert}}$$

Examples:

```
\insert42{abc}
```

The primitive `\insert` is defined in the set `tex`.

The Primitive `\insertpenalties`

`\insertpenalties` is a count register. The primitive `\insertpenalties` is defined in the set `tex`.

The Primitive `\interactionmode`

The formal description of this primitive is the following:

$$\langle \textit{interactionmode} \rangle \\ \rightarrow \text{\texttt{\backslash interactionmode}}$$

Examples:

```
\interactionmode
```

The primitive `\interactionmode` is defined in the set `etex`.

The Primitive `\interlinepenalties`

`\interlinepenalties` is not implemented yet.

The primitive `\interlinepenalties` is defined in the set `etex`.

The Primitive `\interlinepenalty`

`\interlinepenalty` is a count register. The primitive `\interlinepenalty` is defined in the set `tex`.

The Primitive `\javadoc`

The primitive `\javadoc` attaches a definition to a macro or active character. This is done in a similar way as `\def` works. The difference is that the definition has to be provided in form of a Java class.

The general form of this primitive is

$\langle javadef \rangle$
 $\rightarrow \text{\texttt{\code{\javadoc}}} \langle control\ sequence \rangle \langle tokens \rangle$

The $\langle control\ sequence \rangle$ is any macro or active character. If this token is missing or of the wrong type then an error is raised.

The $\langle tokens \rangle$ is any specification of a list of tokens like a constant list enclosed in braces or a toks register. The value of these tokens are taken and interpreted as the name of a Java class. This class is loaded if needed and instantiated. The instance is bound as code to the $\langle control\ sequence \rangle$.

The following example illustrates the use of this primitive:

```
\javadoc\abc{de.dante.extex.interpreter.primitive.Relax}
```

The primitive `\javadoc` is local to the enclosing group as is `\def`. And similar to `\def` the modifier `\global` can be used to make the definition in all groups instead of the current group only. This is shown in the following example:

```
\global\javadoc\abc{de.dante.extex.interpreter.primitive.Relax}
```

Now we come to the Java side of the definition. The class given as $\langle tokens \rangle$ must implement the interface `Code`. The easiest way to achieve this is by declaring a class derived from `AbstractCode`.

```
package my.package;

import de.dante.extex.interpreter.AbstractCode;
import de.dante.extex.interpreter.context.Context;
import de.dante.extex.interpreter.Flags;
import de.dante.extex.interpreter.TokenSource;
import de.dante.extex.typesetter.Typesetter;
import de.dante.util.GeneralException;

class MyPrimitive extends AbstractCode {

    public MyPrimitive(final String name) {
```

```

    super(name);
    // initialization code --if required
}

public boolean execute(final Flags prefix,
                      final Context context,
                      final TokenSource source,
                      final Typesetter typesetter
                      ) {
    // implement the execution behaviour here
    return true;
}
}

```

There is more to say about primitives like how to write expandable primitives or ifs. Those details can be found in section Primitives.

The primitive `\javadoc` is defined in the set `jx`.

The Primitive `\javaload`

The primitive `\javaload` loads a java class and invokes its `init()` method. With this method it is possible to load larger extensions of $\epsilon_X\text{T}_{\text{E}}\text{X}$ in one junk. There is no need to declare each single macro with `\javadoc`.

The general form of this primitive is

$\langle javaload \rangle$
 $\rightarrow \text{\javaload } \langle tokens \rangle$

The $\langle tokens \rangle$ is any specification of a list of tokens like a constant list enclosed in braces or a toks register. The value of these tokens are taken and interpreted as the name of a Java class. This class is loaded if needed, instantiated, and its method `de.dante.extex.interpreter.context.Context, de.dante.extex.typesetter.Typesetter) init()` is invoked. The instantiation requires the empty constructor to be visible.

The following example illustrates the use of this primitive:

```
\javaload{de.dante.extex.extensions.Basic}
```

For the loading of the Java class it is necessary that this Java class implements the interface `Loadable`.

```

package my.package;

import de.dante.extex.interpreter.context.Context;
import de.dante.extex.interpreter.primitives.dynamic.java.Loadable;
import de.dante.extex.typesetter.Typesetter;
import de.dante.util.GeneralException;

```

```
class MyModule implements Loadable {

    public MyModule() {
        super();
        // initialization code --if required
    }

    public void init(final Context context,
                    final Typesetter typesetter
                    ) throws GeneralException {
        // implement the initialization code here
    }
}
```

The primitive `\javaload` is defined in the set `jx`.

The Primitive `\jobname`

The primitive `\jobname` expands to the name of the job currently processed. The job name is usually the name of the first input file. If this can not be determined –e.g. because the input is not coming from a file –then the fallback `texput` is used as default value.

The formal description of this primitive is the following:

$\langle jobname \rangle$
 $\rightarrow \text{\texttt{\textbackslash jobname}}$

Examples:

```
\jobname
```

The primitive `\jobname` is defined in the set `tex`.

The Primitive `\kern`

This primitive produces a horizontal or vertical kerning. This is a (minor) adjustment of the position. The meaning depends on the current mode of the typesetter. In vertical modes it means a vertical adjustment. Otherwise it means a horizontal adjustment.

The formal description of this primitive is the following:

$\langle kern \rangle$
 $\rightarrow \text{\texttt{\textbackslash kern}} \langle dimen \rangle$

Examples:

```
\kern 12pt
```



```
\kern -3mm
```

```
\kern -\dimen123
```

The primitive `\kern` is defined in the set `tex`.

The Primitive `\language`

`\language` is a count register. The primitive `\language` is defined in the set `tex`.

The Primitive `\lastbox`

The formal description of this primitive is the following:

$$\langle lastbox \rangle \rightarrow \backslash lastbox$$

Examples:

```
\lastbox
```

```
\box1=\lastbox
```

The primitive `\lastbox` is defined in the set `tex`.

The Primitive `\lastkern`

Examples:

```
\dimen1=\lastkern
```

The primitive `\lastkern` is defined in the set `tex`.

The Primitive `\lastlinefit`

`\lastlinefit` is not implemented yet.

The primitive `\lastlinefit` is defined in the set `etex`.

The Primitive `\lastnodetype`

Examples:

```
Test\the\lastnodetype
```

The primitive `\lastnodetype` is defined in the set `etex`.

The Primitive `\lastpenalty`

Examples:

```
\count1=\lastpenalty
```

The primitive `\lastpenalty` is defined in the set `tex`.

The Primitive `\lastskip`

`\lastskip` is a skip register. The primitive `\lastskip` is defined in the set `tex`.

The Primitive `\lccode`

The formal description of this primitive is the following:

```
 $\langle lcode \rangle$   
→ \lccode  $\langle \dots \rangle$ 
```

Examples:

```
\lccode ...
```

The primitive `\lccode` is defined in the set `tex`.

The Primitive `\leaders`

The formal description of this primitive is the following:

```
 $\langle leaders \rangle$   
→ \leaders ...
```

Examples:

```
\leaders\hrul\hfill
```

The primitive `\leaders` is defined in the set `tex`.

The Primitive `\left`

The formal description of this primitive is the following:

```
 $\langle left \rangle$   
→ \left
```

Examples:

```
\left
```

The primitive `\left` is defined in the set `tex`.

The Primitive `\lefthyphenmin`

The primitive `\lefthyphenmin` is defined in the set `tex`.

The Primitive `\leftskip`

`\leftskip` is a skip register. The primitive `\leftskip` is defined in the set `tex`.

The Primitive `\leqno`

The formal description of this primitive is the following:

$$\langle span \rangle \rightarrow \backslash leqno$$

Examples:

```
\leqno
```

The primitive `\leqno` is defined in the set `tex`.

The Primitive `\let`

The formal description of this primitive is the following:

$$\langle let \rangle \rightarrow \backslash let \langle control\ sequence \rangle \langle equals \rangle \langle token \rangle$$

Examples:

```
\let\ a=\ b
```

The primitive `\let` is defined in the set `tex`.

The Primitive `\limits`

The formal description of this primitive is the following:

$$\langle limits \rangle \rightarrow \backslash limits$$

Examples:

```
\limits
```

The primitive `\limits` is defined in the set `tex`.

The Primitive `\linepenalty`

`\linepenalty` is a count register. The primitive `\linepenalty` is defined in the set `tex`.

The Primitive `\lineskip`

`\lineskip` is a skip register. The primitive `\lineskip` is defined in the set `tex`.

The Primitive `\lineskiplimit`

`\lineskiplimit` is a dimen register. The primitive `\lineskiplimit` is defined in the set `tex`.

The Primitive `\localbrokenpenalty`

`\localbrokenpenalty` is a count register. The primitive `\localbrokenpenalty` is defined in the set `omega`.

The Primitive `\localinterlinepenalty`

`\localinterlinepenalty` is a count register. The primitive `\localinterlinepenalty` is defined in the set `omega`.

The Primitive `\localleftbox`

`\localleftbox` is not implemented yet.

The primitive `\localleftbox` is defined in the set `omega`.

The Primitive `\localrightbox`

`\localrightbox` is not implemented yet.

The primitive `\localrightbox` is defined in the set `omega`.

The Primitive `\long`

The formal description of this primitive is the following:

$\langle long \rangle$
→ `\long ...`

Examples:

```
\long\def#1{--#1--}
```

The primitive `\long` is defined in the set `tex`.

The Primitive `\looseness`

`\looseness` is a count register. The primitive `\looseness` is defined in the set `tex`.

The Primitive `\lower`

The formal description of this primitive is the following:

$\langle lower \rangle$
 $\rightarrow \text{\texttt{\textbackslash lower}} \langle dimen \rangle \langle box \rangle$

Examples:

```
\lower 2em \hbox{abc}
```

```
\lower -1pt \hbox to 120pt {abc}
```

```
\lower 2mm \hbox spread 12pt {abc}
```

The primitive `\lower` is defined in the set `tex`.

The Primitive `\lowercase`

The formal description of this primitive is the following:

$\langle lowercase \rangle$
 $\rightarrow \text{\texttt{\textbackslash lowercase}} \langle \dots \rangle$

Examples:

```
\lowercase ...
```

The primitive `\lowercase` is defined in the set `tex`.

The Primitive `\mag`

The formal description of this primitive is the following:

$\langle mag \rangle$
 $\rightarrow \text{\texttt{\textbackslash mag}}$

Examples:

```
\count23=-456
```

The primitive `\mag` is defined in the set `tex`.

The Primitive `\mark`

The formal description of this primitive is the following:

`\mark ...`

Examples:

```
\mark{abc}
```

The primitive `\mark` is defined in the set `tex`.

The Primitive `\marks`

The formal description of this primitive is the following:

`\marks ...`

Examples:

```
\marks123{abc}
```

The primitive `\marks` is defined in the set `etex`.

The Primitive `\mathaccent`

The formal description of this primitive is the following:

$\langle mathaccent \rangle$
 $\rightarrow \text{\code{\mathaccent}}$

Examples:

```
\mathaccent
```

The primitive `\mathaccent` is defined in the set `tex`.

The Primitive `\mathbin`

The formal description of this primitive is the following:

$\langle mathbin \rangle$
 $\rightarrow \text{\code{\mathbin}}$

Examples:

```
\mathbin
```

The primitive `\mathbin` is defined in the set `tex`.

The Primitive `\mathchar`

The primitive `\mathchar` inserts a mathematical character consisting of a math class and a character code into the current math list. This is supposed to work in math mode only.

The formal description of this primitive is the following:

`\mathchar` ...

Examples:

```
\mathchar"041
```

```
\mathchar{ordinary}0 ‘A
```

The primitive `\mathchar` is defined in the set `tex`.

The Primitive `\mathchardef`

The formal description of this primitive is the following:

`\mathchardef` ...

Examples:

```
\mathchardef\alpha ...
```

The primitive `\mathchardef` is defined in the set `tex`.

The Primitive `\mathchoice`

The formal description of this primitive is the following:

$\langle\mathchoice\rangle$
 $\rightarrow \mathchoice$

Examples:

```
\mathchoice{d}{t}{s}{ss}
```

The primitive `\mathchoice` is defined in the set `tex`.

The Primitive `\mathclose`

The formal description of this primitive is the following:

$$\langle\mathclose\rangle \rightarrow \backslash\mathclose$$

Examples:

```
\mathclose
```

The primitive `\mathclose` is defined in the set `tex`.

The Primitive `\mathcode`

The formal description of this primitive is the following:

$$\backslash\mathcode \dots$$

Examples:

```
\mathcode \dots
```

The primitive `\mathcode` is defined in the set `tex`.

The Primitive `\mathdir`

`\mathdir` is not implemented yet.

The primitive `\mathdir` is defined in the set `omega`.

The Primitive `\mathinner`

The formal description of this primitive is the following:

$$\langle\mathinner\rangle \rightarrow \backslash\mathinner \langle\math block\rangle$$

Examples:

```
\mathinner{a^b}
```

The primitive `\mathinner` is defined in the set `tex`.

The Primitive `\mathop`

The formal description of this primitive is the following:

$$\langle \mathit{mathop} \rangle$$

$$\rightarrow \backslash \mathit{mathop}$$

Examples:

```
\mathop
```

The primitive `\mathop` is defined in the set `tex`.

The Primitive `\mathopen`

The formal description of this primitive is the following:

$$\langle \mathit{mathopen} \rangle$$

$$\rightarrow \backslash \mathit{mathopen}$$

Examples:

```
\mathopen
```

The primitive `\mathopen` is defined in the set `tex`.

The Primitive `\mathord`

The formal description of this primitive is the following:

$$\langle \mathit{mathord} \rangle$$

$$\rightarrow \backslash \mathit{mathord}$$

Examples:

```
\mathord
```

The primitive `\mathord` is defined in the set `tex`.

The Primitive `\mathpunct`

The formal description of this primitive is the following:

$$\langle \mathit{mathpunct} \rangle$$

$$\rightarrow \backslash \mathit{mathpunct}$$

Examples:

```
\mathpunct
```

The primitive `\mathpunct` is defined in the set `tex`.

The Primitive `\mathrel`

The formal description of this primitive is the following:

$$\langle \mathit{mathrel} \rangle \\ \rightarrow \quad \backslash\mathit{mathrel}$$

Examples:

```
\mathrel
```

The primitive `\mathrel` is defined in the set `tex`.

The Primitive `\mathsurround`

`\mathsurround` is a `dimen` register. The primitive `\mathsurround` is defined in the set `tex`.

The Primitive `\maxdeadcycles`

`\maxdeadcycles` is a `count` register. The primitive `\maxdeadcycles` is defined in the set `tex`.

The Primitive `\maxdepth`

`\maxdepth` is a `dimen` register. The primitive `\maxdepth` is defined in the set `tex`.

The Primitive `\meaning`

The formal description of this primitive is the following:

$$\langle \mathit{meaning} \rangle \\ \rightarrow \quad \backslash\mathit{meaning} \langle \mathit{token} \rangle$$

Examples:

```
\meaning a
```

The primitive `\meaning` is defined in the set `tex`.

The Primitive `\medmuskip`

The primitive `\medmuskip` is defined in the set `tex`.

The Primitive `\message`

The primitive `\message` is defined in the set `tex`.

The Primitive `\middle`

The formal description of this primitive is the following:

$$\langle span \rangle \rightarrow \backslash middle$$

Examples:

```
\middle
```

The primitive `\middle` is defined in the set `etex`.

The Primitive `\mkern`

The formal description of this primitive is the following:

$$\langle mkern \rangle \rightarrow \backslash mkern$$

Examples:

```
\mkern
```

The primitive `\mkern` is defined in the set `tex`.

The Primitive `\month`

`\month` is a count register. The primitive `\month` is defined in the set `tex`.

The Primitive `\moveleft`

The formal description of this primitive is the following:

$$\langle moveleft \rangle \rightarrow \backslash moveleft \langle dimen \rangle \langle box \rangle$$

Examples:

```
\moveleft 2em \hbox{abc}
```

```
\moveleft -1pt \hbox to 120pt {abc}
```

```
\moveleft 2mm \hbox spread 12pt {abc}
```

The primitive `\moveleft` is defined in the set `tex`.

The Primitive `\moveright`

The formal description of this primitive is the following:

$\langle moveright \rangle$
 $\rightarrow \text{\code{\moveright}} \langle dimen \rangle \langle box \rangle$

The color from the typographic context is taken as foreground color for the rule. The default color is black.

Examples:

```
\moveright 2em \hbox{abc}
```

```
\moveright -1pt \hbox to 120pt {abc}
```

```
\moveright 2mm \hbox spread 12pt {abc}
```

The primitive `\moveright` is defined in the set `tex`.

The Primitive `\mskip`

The formal description of this primitive is the following:

$\langle mskip \rangle$
 $\rightarrow \text{\code{\mskip}}$

Examples:

```
\mskip 12mu plus 3mu minus 4 mu
```

The primitive `\mskip` is defined in the set `tex`.

The Primitive `\muexpr`

`\muexpr` is not implemented yet.

The primitive `\muexpr` is defined in the set `etex`.

The Primitive `\multiply`

This primitive implements an assignment. The variable given as next tokens is multiplied by the quantity given after the optional `by`.

The formal description of this primitive is the following:

$\langle multiply \rangle$
 $\rightarrow \backslash multiply \langle multiplyable \rangle$
 $\langle multiplyable \rangle$
 $\rightarrow \langle integer\ variable \rangle \langle optional\ by \rangle \langle 8\text{-bit}\ number \rangle$
 $| \quad \langle dimen\ variable \rangle \langle optional\ by \rangle \langle 8\text{-bit}\ number \rangle$
 $| \quad \langle glue\ variable \rangle \langle optional\ by \rangle \langle 8\text{-bit}\ number \rangle$
 $| \quad \langle muglue\ variable \rangle \langle optional\ by \rangle \langle 8\text{-bit}\ number \rangle$
 $\langle optional\ by \rangle$
 $\rightarrow [by]$
 $| \quad \langle optional\ spaces \rangle$

Examples:

```
\multiply\count12 345
```

```
\multiply\count12 by -345
```

The primitive `\multiply` is defined in the set `tex`.

The Primitive `\muskip`

The primitive `\muskip` is defined in the set `tex`.

The Primitive `\muskipdef`

The formal description of this primitive is the following:

$\backslash muskipdef \langle control\ sequence \rangle \langle equals \rangle \langle 8\text{-bit}\ number \rangle$

Examples:

```
\muskipdef\abc=45
```

```
\muskipdef\abc 33
```

The primitive `\muskipdef` is defined in the set `tex`.

The Primitive `\namespace`

The formal description of this primitive is the following:

$\langle namespace \rangle$
 $\rightarrow \backslash namespace \langle replacement\ text \rangle$

4. The Macro Language of $\epsilon\chi\TeX$

Examples:

```
\namespace{org.dante.dtk}
```

The primitive `\namespace` is defined in the set `namespace`.

The Primitive `\nativedef`

The primitive `\nativedef` attaches a definition to a macro or active character. This is done in a similar way as `\def` works. The difference is that the definition has to be provided in form of a Java class which glues in native code.

The general form of this primitive is

$\langle nativedef \rangle$
 $\rightarrow \text{\texttt{\textbackslash nativedef}} \langle control\ sequence \rangle \langle name \rangle$

The $\langle control\ sequence \rangle$ is any macro or active character. If this token is missing or of the wrong type then an error is raised.

The $\langle name \rangle$ is any specification of a list of tokens like a constant list enclosed in braces or a toks register. The value of these tokens are taken and resolved via the configuration. This appropriate class is loaded if needed and instantiated. The instance is bound as code to the $\langle control\ sequence \rangle$.

The primitive `\javadoc` is local to the enclosing group as is `\def`. And similar to `\def` the modifier `\global` can be used to make the definition in all groups instead of the current group only.

The primitive `\nativedef` is defined in the set `native`.

The Primitive `\nativeload`

The general form of this primitive is

$\langle nativeload \rangle$
 $\rightarrow \text{\texttt{\textbackslash nativeload}} \langle type \rangle \langle tokens \rangle$

The primitive `\nativeload` is defined in the set `native`.

The Primitive `\naturaldir`

`\naturaldir` is not implemented yet.

The primitive `\naturaldir` is defined in the set `omega`.

The Primitive `\newlinechar`

`\newlinechar` is a count register. The primitive `\newlinechar` is defined in the set `tex`.

The Primitive `\noalign`

The formal description of this primitive is the following:

$$\langle noalign \rangle \\ \rightarrow \quad \backslash noalign$$

Examples:

```
\cr\noalign
```

The primitive `\noalign` is defined in the set `tex`.

The Primitive `\noboundary`

The formal description of this primitive is the following:

$$\langle noboundary \rangle \\ \rightarrow \quad \backslash \backslash$$

Examples:

```
\backslash
```

The primitive `\noboundary` is defined in the set `tex`.

The Primitive `\noDefaultInputMode`

`\noDefaultInputMode` is not implemented yet.

The primitive `\noDefaultInputMode` is defined in the set `omega`.

The Primitive `\noDefaultInputTranslation`

`\noDefaultInputTranslation` is not implemented yet.

The primitive `\noDefaultInputTranslation` is defined in the set `omega`.

The Primitive `\noDefaultOutputMode`

`\noDefaultOutputMode` is not implemented yet.

The primitive `\noDefaultOutputMode` is defined in the set `omega`.

The Primitive `\noDefaultOutputTranslation`

`\noDefaultOutputTranslation` is not implemented yet.

The primitive `\noDefaultOutputTranslation` is defined in the set `omega`.

The Primitive `\noexpand`

The formal description of this primitive is the following:

$\langle noexpand \rangle$
 $\rightarrow \text{\texttt{\noexpand}}$

Examples:

```
\noexpand
```

The primitive `\noexpand` is defined in the set `tex`.

The Primitive `\noindent`

The formal description of this primitive is the following:

$\langle noindent \rangle$
 $\rightarrow \text{\texttt{\noindent}}$

Examples:

```
\noindent
```

The primitive `\noindent` is defined in the set `tex`.

The Primitive `\nolimits`

The formal description of this primitive is the following:

$\langle nolimits \rangle$
 $\rightarrow \text{\texttt{\nolimits}}$

Examples:

```
\nolimits
```

The primitive `\nolimits` is defined in the set `tex`.

The Primitive `\nonscript`

The primitive can be used in math modes only. It cancels following glue if the current style is script style or scriptscript style.

The formal description of this primitive is the following:

$\langle nonscript \rangle$
 $\rightarrow \text{\texttt{\nonscript}}$

Examples:

```
\nonscript
```

The primitive `\nonscript` is defined in the set `tex`.

The Primitive `\nonstopmode`

The formal description of this primitive is the following:

$$\langle nonstopmode \rangle \rightarrow \backslash nonstopmode$$

Examples:

```
\nonstopmode
```

The primitive `\nonstopmode` is defined in the set `tex`.

The Primitive `\nulldelimiterspace`

`\nulldelimiterspace` is a dimen register. The primitive `\nulldelimiterspace` is defined in the set `tex`.

The Primitive `\nullfont`

The formal description of this primitive is the following:

```
\nullfont
```

Examples:

```
\font123=\nullfont
```

The primitive `\nullfont` is defined in the set `tex`.

The Primitive `\nullocplist`

`\nullocplist` is not implemented yet.

The primitive `\nullocplist` is defined in the set `omega`.

The Primitive `\number`

The formal description of this primitive is the following:

$$\langle number \rangle \rightarrow \backslash number \langle \dots \rangle$$

Examples:

```
\number ...
```

The primitive `\number` is defined in the set `tex`.

The Primitive `\numexpr`

`\numexpr` is not implemented yet.

The primitive `\numexpr` is defined in the set `etex`.

The Primitive `\ocp`

`\ocp` is not implemented yet.

The primitive `\ocp` is defined in the set `omega`.

The Primitive `\ocplist`

`\ocplist` is not implemented yet.

The primitive `\ocplist` is defined in the set `omega`.

The Primitive `\odelimiter`

`\odelimiter` is not implemented yet.

The primitive `\odelimiter` is defined in the set `omega`.

The Primitive `\omathaccent`

`\omathaccent` is not implemented yet.

The primitive `\omathaccent` is defined in the set `omega`.

The Primitive `\omathchar`

`\omathchar` is not implemented yet.

The primitive `\omathchar` is defined in the set `omega`.

The Primitive `\omathchardef`

`\omathchardef` is not implemented yet.

The primitive `\omathchardef` is defined in the set `omega`.

The Primitive `\omathcode`

`\omathcode` is not implemented yet.

The primitive `\omathcode` is defined in the set `omega`.

The Primitive `\omathdelcode`

`\omathdelcode` is not implemented yet.

The primitive `\omathdelcode` is defined in the set `omega`.

The Primitive `\omit`

The formal description of this primitive is the following:

$$\langle omit \rangle \\ \rightarrow \texttt{\backslash omit}$$

Examples:

```
\omit 1
```

The primitive `\omit` is defined in the set `tex`.

The Primitive `\openin`

The primitive `\openin` is defined in the set `tex`.

The Primitive `\openout`

The primitive `\openout` is defined in the set `tex`.

The Primitive `\or`

$$\langle or \rangle \\ \rightarrow \texttt{\backslash ifcase ... \backslash or ... \backslash fi}$$

The primitive `\or` is defined in the set `tex`.

The Primitive `\oradical`

`\oradical` is not implemented yet.

The primitive `\oradical` is defined in the set `omega`.

The Primitive `\outer`

The formal description of this primitive is the following:

$\langle outer \rangle$
 $\rightarrow \texttt{\backslash outer} \dots$

Examples:

```
\outer\def#1{--#1--}
```

The primitive `\outer` is defined in the set `tex`.

The Primitive `\output`

`\output` is a toks register. The primitive `\output` is defined in the set `tex`.

The Primitive `\OutputMode`

`\OutputMode` is not implemented yet.

The primitive `\OutputMode` is defined in the set `omega`.

The Primitive `\outputpenalty`

`\outputpenalty` is a count register. The primitive `\outputpenalty` is defined in the set `tex`.

The Primitive `\OutputTranslation`

`\OutputTranslation` is not implemented yet.

The primitive `\OutputTranslation` is defined in the set `omega`.

The Primitive `\over`

The formal description of this primitive is the following:

$\langle over \rangle$
 $\rightarrow \texttt{\backslash over}$

Examples:

```
a \over b
```

The primitive `\over` is defined in the set `tex`.

The Primitive `\overfullrule`

`\overfullrule` is a dimen register. The primitive `\overfullrule` is defined in the set `tex`.

The Primitive `\overline`

The formal description of this primitive is the following:

$$\langle span \rangle \rightarrow \text{\code{\overline}}$$

Examples:

```
\overline
```

The primitive `\overline` is defined in the set `tex`.

The Primitive `\overwithdelims`

The formal description of this primitive is the following:

$$\langle overwithdelims \rangle \rightarrow \text{\code{\overwithdelims}}$$

Examples:

```
\overwithdelims
```

The primitive `\overwithdelims` is defined in the set `tex`.

The Primitive `\pagedepth`

`\pagedepth` is a dimen register. The primitive `\pagedepth` is defined in the set `tex`.

The Primitive `\pagedir`

`\pagedir` is not implemented yet.

The primitive `\pagedir` is defined in the set `omega`.

The Primitive `\pagedirHL`

`\pagedirHL` is not implemented yet.

The primitive `\pagedirHL` is defined in the set `omega`.

The Primitive `\pagedirHR`

`\pagedirHR` is not implemented yet.

The primitive `\pagedirHR` is defined in the set `omega`.

The Primitive `\pagediscarts`

`\pagediscarts` is not implemented yet.

The primitive `\pagediscarts` is defined in the set `etex`.

The Primitive `\pagefillllstretch`

`\pagefillllstretch` is a dimen register. The primitive `\pagefillllstretch` is defined in the set `tex`.

The Primitive `\pagefillstretch`

`\pagefillstretch` is a dimen register. The primitive `\pagefillstretch` is defined in the set `tex`.

The Primitive `\pagefilstretch`

`\pagefilstretch` is a dimen register. The primitive `\pagefilstretch` is defined in the set `tex`.

The Primitive `\pagegoal`

`\pagegoal` is a dimen register. The primitive `\pagegoal` is defined in the set `tex`.

The Primitive `\pageshrink`

`\pageshrink` is a dimen register. The primitive `\pageshrink` is defined in the set `tex`.

The Primitive `\pagestretch`

`\pagestretch` is a dimen register. The primitive `\pagestretch` is defined in the set `tex`.

The Primitive `\pagetotal`

`\pagetotal` is a dimen register. The primitive `\pagetotal` is defined in the set `tex`.

The Primitive `\par`

The formal description of this primitive is the following:

$$\langle par \rangle \rightarrow \backslash par$$

Examples:

The primitive `\par` is defined in the set `tex`.

The Primitive `\parfillskip`

`\parfillskip` is a skip register. The primitive `\parfillskip` is defined in the set `tex`.

The Primitive `\parindent`

`\parindent` is a dimen register. The primitive `\parindent` is defined in the set `tex`.

The Primitive `\parshape`

The primitive `\parshape` is a declaration of the shape of the paragraph. With its help it is possible to control the left and right margin of the current paragraph.

The formal description of this primitive is the following:

$$\langle parshape \rangle \rightarrow \backslash parshape \langle 8\text{-bit number} \rangle \dots$$

Examples:

```
\parshape 3 20pt \linewidth
          20pt \linewidth
          0pt \linewidth
```

```
\parshape 0
```

`\parshape` acts as special register which can be queried. It returns the size of the current parshape specification or 0 if none is present.

The primitive `\parshape` is defined in the set `tex`.

The Primitive `\parshapedimen`

`\parshapedimen` is not implemented yet.

The primitive `\parshapedimen` is defined in the set `etex`.

The Primitive `\parshapeindent`

`\parshapeindent` is not implemented yet.

The primitive `\parshapeindent` is defined in the set `etex`.

The Primitive `\parshapelength`

`\parshapelength` is not implemented yet.

The primitive `\parshapelength` is defined in the set `etex`.

The Primitive `\parskip`

`\parskip` is a skip register. The primitive `\parskip` is defined in the set `tex`.

The Primitive `\patterns`

The formal description of this primitive is the following:

$$\langle patterns \rangle \rightarrow \texttt{\backslash patterns} \langle patterns \rangle$$

Examples:

```
\patterns{.ach4 .ad4der .af1t}
```

The primitive `\patterns` is defined in the set `tex`.

The Primitive `\pausing`

`\pausing` is a count register. The primitive `\pausing` is defined in the set `tex`.

The Primitive `\penalty`

This primitive inserts penalty into the current node list. In vertical mode the page builder is also invoked.

A penalty of 10000 or more will inhibit a break at this position. A penalty of -10000 or less will force a break at this position.

The formal description of this primitive is the following:

$\langle \textit{penalty} \rangle$
 $\rightarrow \backslash \textit{penalty} \langle 8\text{-bit number} \rangle$

Examples:

```
\penalty 123
```

```
\penalty -456
```

```
\penalty -\count254
```

The primitive `\penalty` is defined in the set `tex`.

The Primitive `\popocplist`

`\popocplist` is not implemented yet.

The primitive `\popocplist` is defined in the set `omega`.

The Primitive `\postdisplaypenalty`

`\postdisplaypenalty` is a count register. The primitive `\postdisplaypenalty` is defined in the set `tex`.

The Primitive `\predisplaydirection`

`\predisplaydirection` is not implemented yet.

The primitive `\predisplaydirection` is defined in the set `etex`.

The Primitive `\predisplaypenalty`

`\predisplaypenalty` is a count register. The primitive `\predisplaypenalty` is defined in the set `tex`.

The Primitive `\predisplaysize`

`\predisplaysize` is a dimen register. The primitive `\predisplaysize` is defined in the set `tex`.

The Primitive `\pretolerance`

`\pretolerance` is a count register. The primitive `\pretolerance` is defined in the set `tex`.

The Primitive `\prevdepth`

The formal description of this primitive is the following:

$$\langle prevdepth \rangle \\ \rightarrow \text{\code{\prevdepth}} \dots$$

Examples:

```
\prevdepth ...
```

The primitive `\prevdepth` is defined in the set `tex`.

The Primitive `\prevgraf`

The formal description of this primitive is the following:

$$\langle prevgraf \rangle \\ \rightarrow \text{\code{\prevgraf}}$$

Examples:

```
\prevgraf
```

The primitive `\prevgraf` is defined in the set `tex`.

The Primitive `\protected`

The formal description of this primitive is the following:

$$\langle protected \rangle \\ \rightarrow \text{\code{\protected}}$$

Examples:

```
\protected\def\abc{123}
```

The primitive `\protected` is defined in the set `etex`.

The Primitive `\pushocplist`

`\pushocplist` is not implemented yet.

The primitive `\pushocplist` is defined in the set `omega`.

The Primitive `\radical`

The formal description of this primitive is the following:

$$\langle radical \rangle \rightarrow \backslash radical$$

Examples:

```
\radical
```

The primitive `\radical` is defined in the set `tex`.

The Primitive `\raise`

The formal description of this primitive is the following:

$$\langle raise \rangle \rightarrow \backslash raise \langle dimen \rangle \langle box \rangle$$

Examples:

```
\raise 2em \hbox{abc}
```

```
\raise -1pt \hbox to 120pt {abc}
```

```
\raise 2mm \hbox spread 12pt {abc}
```

The primitive `\raise` is defined in the set `tex`.

The Primitive `\read`

The formal description of this primitive is the following:

$$\langle read \rangle \rightarrow \backslash read \langle read \rangle \text{ to } \langle control sequence \rangle$$

The primitive `\read` is defined in the set `tex`.

The Primitive `\readline`

`\readline` is not implemented yet.

The primitive `\readline` is defined in the set `etex`.

The Primitive `\relax`

This primitive simply does nothing. It acts as a no-op for the $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ macro language.

The formal description of this primitive is the following:

$$\langle relax \rangle \\ \rightarrow \quad \backslash relax$$

Examples:

```
\relax
```

```
\the\count123\relax456
```

The primitive `\relax` is defined in the set `tex`.

The Primitive `\relpenalty`

`\relpenalty` is a count register. The primitive `\relpenalty` is defined in the set `tex`.

The Primitive `\removebeforeocplist`

`\removebeforeocplist` is not implemented yet.

The primitive `\removebeforeocplist` is defined in the set `omega`.

The Primitive `\right`

The formal description of this primitive is the following:

$$\langle span \rangle \\ \rightarrow \quad \backslash right$$

Examples:

```
\right
```

The primitive `\right` is defined in the set `tex`.

The Primitive `\righthyphenmin`

The primitive `\righthyphenmin` is defined in the set `tex`.

The Primitive `\rightskip`

`\rightskip` is a skip register. The primitive `\rightskip` is defined in the set `tex`.

The Primitive `\romannumeral`

The formal description of this primitive is the following:

$$\langle \textit{romannumeral} \rangle \\ \rightarrow \texttt{\backslash romannumeral} \langle \textit{number} \rangle$$

Examples:

```
\romannumeral\count1
```

```
\romannumeral 2004
```

The primitive `\romannumeral` is defined in the set `tex`.

The Primitive `\savingshyphcodes`

`\savingshyphcodes` is not implemented yet.

The primitive `\savingshyphcodes` is defined in the set `etex`.

The Primitive `\savingsdiscarts`

`\savingsdiscarts` is not implemented yet.

The primitive `\savingsdiscarts` is defined in the set `etex`.

The Primitive `\scantokens`

`\scantokens` is not implemented yet.

The primitive `\scantokens` is defined in the set `etex`.

The Primitive `\scriptfont`

The primitive `\scriptfont` is defined in the set `tex`.

The Primitive `\scriptscriptfont`

The primitive `\scriptscriptfont` is defined in the set `tex`.

The Primitive `\scriptscriptstyle`

The formal description of this primitive is the following:

$$\langle \textit{scriptscriptstyle} \rangle \\ \rightarrow \texttt{\backslash scriptscriptstyle}$$

4. The Macro Language of $\epsilon\chi\TeX$

Examples:

```
\scriptscriptstyle
```

The primitive `\scriptscriptstyle` is defined in the set `tex`.

The Primitive `\scriptspace`

`\scriptspace` is a dimen register. The primitive `\scriptspace` is defined in the set `tex`.

The Primitive `\scriptstyle`

The formal description of this primitive is the following:

$\langle scriptstyle \rangle$
→ `\scriptstyle`

Examples:

```
\scriptstyle
```

The primitive `\scriptstyle` is defined in the set `tex`.

The Primitive `\scrollmode`

The formal description of this primitive is the following:

$\langle scrollmode \rangle$
→ `\scrollmode`

Examples:

```
\scrollmode
```

The primitive `\scrollmode` is defined in the set `tex`.

The Primitive `\setbox`

The formal description of this primitive is the following:

$\langle setbox \rangle$
→ `\setbox` $\langle 8\text{-bit number} \rangle \dots$

Examples:

```
\setbox0\hbox{abc}
```

The primitive `\setbox` is defined in the set `tex`.

The Primitive `\setlanguage`

The formal description of this primitive is the following:

$$\langle \textit{setlanguage} \rangle \\ \rightarrow \texttt{\setlanguage} \langle \textit{number} \rangle$$

Examples:

```
\setlanguage2
```

The primitive `\setlanguage` is defined in the set `tex`.

The Primitive `\sfcode`

The formal description of this primitive is the following:

$$\langle \textit{sfcode} \rangle \\ \rightarrow \texttt{\sfcode} \dots$$

Examples:

```
\sfcode ...
```

The primitive `\sfcode` is defined in the set `tex`.

The Primitive `\shipout`

The primitive `\shipout` takes a box and send the contents of the box to the document writer.

In addition the count register `\deadcycles` is reset to 0. This count register is used to break out of infinite loops when no material is shipped out in the output routine.

The formal description of this primitive is the following:

$$\langle \textit{shipout} \rangle \\ \rightarrow \texttt{\shipout} \langle \textit{box} \rangle$$

Examples:

```
\shipout\box255
```

The primitive `\shipout` is defined in the set `tex`.

The Primitive `\show`

The formal description of this primitive is the following:

$$\langle show \rangle \rightarrow \backslash\text{show} \langle token \rangle$$

Examples:

```
\show\abc
```

The primitive `\show` is defined in the set `tex`.

The Primitive `\showbox`

The formal description of this primitive is the following:

$$\langle showbox \rangle \rightarrow \backslash\text{showbox} \langle 8\text{-bit number} \rangle$$

Examples:

```
\showbox 1
```

The primitive `\showbox` is defined in the set `tex`.

The Primitive `\showboxbreadth`

`\showboxbreadth` is a count register. The primitive `\showboxbreadth` is defined in the set `tex`.

The Primitive `\showboxdepth`

`\showboxdepth` is a count register. The primitive `\showboxdepth` is defined in the set `tex`.

The Primitive `\showgroups`

`\showgroups` is not implemented yet.

The primitive `\showgroups` is defined in the set `etex`.

The Primitive `\showlists`

The formal description of this primitive is the following:

$$\langle \textit{showlists} \rangle \\ \rightarrow \quad \backslash\textit{showlists}$$

Examples:

```
\showlists 1
```

The primitive `\showlists` is defined in the set `tex`.

The Primitive `\showthe`

The primitive `\showthe` is defined in the set `tex`.

The Primitive `\showtokens`

`\showtokens` is not implemented yet.

The primitive `\showtokens` is defined in the set `etex`.

The Primitive `\skewchar`

The formal description of this primitive is the following:

$$\backslash\textit{skewchar} \langle \textit{font} \rangle \langle \textit{equals} \rangle \langle 8\text{-bit number} \rangle$$

Examples:

```
\skewchar\font=123
```

Incompatibility

The TeXbook gives no indication on how the primitive should react for negative values – except -1. The implementation of $\mathrm{T}_E\mathrm{X}$ allows to store and retrieve arbitrary negative values. This behaviour of $\mathrm{T}_E\mathrm{X}$ is not preserved in $\epsilon_X\mathrm{T}_E\mathrm{X}$.

The primitive `\skewchar` is defined in the set `tex`.

The Primitive `\skip`

The primitive `\skip` is defined in the set `tex`.

The Primitive `\skipdef`

The formal description of this primitive is the following:

`\skipdef` $\langle control\ sequence \rangle$ $\langle equals \rangle$ $\langle 8\text{-bit}\ number \rangle$

Examples:

```
\skipdef\abc=45
```

```
\skipdef\abc 33
```

The primitive `\skipdef` is defined in the set `tex`.

The Primitive `\spacefactor`

The formal description of this primitive is the following:

$\langle spacefactor \rangle$
 \rightarrow `\spacefactor` ...

Examples:

```
\spacefactor ...
```

The primitive `\spacefactor` is defined in the set `tex`.

The Primitive `\spaceskip`

`\spaceskip` is a skip register. The primitive `\spaceskip` is defined in the set `tex`.

The Primitive `\span`

The formal description of this primitive is the following:

$\langle span \rangle$
 \rightarrow `\span`

Examples:

```
\span 1
```

The primitive `\span` is defined in the set `tex`.

The Primitive `\special`

This primitive sends a string to the backend driver. The argument is a balanced block of text which is expanded and translated into a string. The string is given in a `SpecialNode` to the typesetter for passing it down.

The formal description of this primitive is the following:

$$\langle special \rangle \rightarrow \backslash special \langle general\ text \rangle$$

Examples:

```
\special{hello world}
```

```
\special{ps: \abc}
```

For several backend drivers for T_EX a quasi-standard has emerged which uses a prefix ended by a colon to indicate the backend driver the special is targeted at.

The primitive `\special` is defined in the set `tex`.

The Primitive `\splitbotmark`

The formal description of this primitive is the following:

```
\splitbotmark ...
```

Examples:

```
\splitbotmark ...
```

The primitive `\splitbotmark` is defined in the set `tex`.

The Primitive `\splitbotmarks`

`\splitbotmarks` is not implemented yet.

The primitive `\splitbotmarks` is defined in the set `etex`.

The Primitive `\splitdiscarts`

`\splitdiscarts` is not implemented yet.

The primitive `\splitdiscarts` is defined in the set `etex`.

The Primitive `\splitfirstmark`

The formal description of this primitive is the following:

`\splitfirstmark ...`

Examples:

```
\splitfirstmark ...
```

The primitive `\splitfirstmark` is defined in the set `tex`.

The Primitive `\splitfirstmarks`

`\splitfirstmarks` is not implemented yet.

The primitive `\splitfirstmarks` is defined in the set `etex`.

The Primitive `\splitmaxdepth`

`\splitmaxdepth` is a `dimen` register. The primitive `\splitmaxdepth` is defined in the set `tex`.

The Primitive `\splittopskip`

`\splittopskip` is a `skip` register. The primitive `\splittopskip` is defined in the set `tex`.

The Primitive `\string`

This primitive takes the next unexpanded token. If this token is a control sequence –and no active character –then the value of `escapechar` followed by the characters from the name of the control sequence. Otherwise it is a single character token containing the character code of the token.

The formal description of this primitive is the following:

$$\langle string \rangle \rightarrow \texttt{\string} \langle token \rangle$$

Examples:

```
\string ...
```

The primitive `\string` is defined in the set `tex`.

The Primitive `\tabskip`

`\tabskip` is a `skip` register. The primitive `\tabskip` is defined in the set `tex`.

The Primitive `\textdir`

`\textdir` is not implemented yet.

The primitive `\textdir` is defined in the set `omega`.

The Primitive `\textfont`

The primitive `\textfont` is defined in the set `tex`.

The Primitive `\textstyle`

The formal description of this primitive is the following:

$$\langle \textit{textstyle} \rangle \\ \rightarrow \texttt{\textbackslash textstyle}$$

Examples:

```
\textstyle
```

The primitive `\textstyle` is defined in the set `tex`.

The Primitive `\TeXeTstate`

`\TeXeTstate` is a count register. The primitive `\TeXeTstate` is defined in the set `etex`.

The Primitive `\the`

The formal description of this primitive is the following:

$$\langle \textit{the} \rangle \\ \rightarrow \texttt{\textbackslash the} \langle \textit{internal quantity} \rangle$$

Examples:

```
\the\count123
```

The primitive `\the` is defined in the set `tex`.

The Primitive `\thickmuskip`

The primitive `\thickmuskip` is defined in the set `tex`.

The Primitive `\thinmuskip`

The primitive `\thinmuskip` is defined in the set `tex`.

The Primitive `\time`

`\time` is a count register. The primitive `\time` is defined in the set `tex`.

The Primitive `\toks`

The primitive `\toks` is defined in the set `tex`.

The Primitive `\toksdef`

The formal description of this primitive is the following:

`\toksdef` *<control sequence>* *<equals>* *<8-bit number>*

Examples:

```
\toksdef\abc=45
```

```
\toksdef\abc 33
```

The primitive `\toksdef` is defined in the set `tex`.

The Primitive `\tolerance`

`\tolerance` is a count register. The primitive `\tolerance` is defined in the set `tex`.

The Primitive `\topmark`

The formal description of this primitive is the following:

`\topmark` ...

Examples:

```
\topmark ...
```

The primitive `\topmark` is defined in the set `tex`.

The Primitive `\topmarks`

`\topmarks` is not implemented yet.

The primitive `\topmarks` is defined in the set `etex`.

The Primitive `\topskip`

`\topskip` is a skip register. The primitive `\topskip` is defined in the set `tex`.

The Primitive `\tracingassigns`

`\tracingassigns` is a count register. The primitive `\tracingassigns` is defined in the set `etex`.

The Primitive `\tracingcommands`

`\tracingcommands` is a count register. The primitive `\tracingcommands` is defined in the set `tex`.

The Primitive `\tracinggroups`

`\tracinggroups` is a count register. The primitive `\tracinggroups` is defined in the set `etex`.

The Primitive `\tracingifs`

`\tracingifs` is a count register. The primitive `\tracingifs` is defined in the set `etex`.

The Primitive `\tracinglostchars`

`\tracinglostchars` is a count register. The primitive `\tracinglostchars` is defined in the set `tex`.

The Primitive `\tracingmacros`

`\tracingmacros` is a count register. The primitive `\tracingmacros` is defined in the set `tex`.

The Primitive `\tracingnesting`

`\tracingnesting` is a count register. The primitive `\tracingnesting` is defined in the set `etex`.

The Primitive `\tracingonline`

`\tracingonline` is a count register. The primitive `\tracingonline` is defined in the set `tex`.

The Primitive `\tracingoutput`

`\tracingoutput` is a count register. The primitive `\tracingoutput` is defined in the set `tex`.

The Primitive `\tracingpages`

`\tracingpages` is a count register. The primitive `\tracingpages` is defined in the set `tex`.

The Primitive `\tracingparagraphs`

`\tracingparagraphs` is a count register. The primitive `\tracingparagraphs` is defined in the set `tex`.

The Primitive `\tracingrestores`

`\tracingrestores` is a count register. The primitive `\tracingrestores` is defined in the set `tex`.

The Primitive `\tracingscantokens`

`\tracingscantokens` is a count register. The primitive `\tracingscantokens` is defined in the set `etex`.

The Primitive `\tracingstats`

`\tracingstats` is a count register. The primitive `\tracingstats` is defined in the set `tex`.

The Primitive `\uccode`

The formal description of this primitive is the following:

$$\langle uccode \rangle \rightarrow \text{\code \uccode} \langle \dots \rangle$$

Examples:

```
\uccode ...
```

The primitive `\uccode` is defined in the set `tex`.

The Primitive `\uchyph`

`\uchyph` is a count register. The primitive `\uchyph` is defined in the set `tex`.

The Primitive `\underline`

The formal description of this primitive is the following:

$$\langle span \rangle \rightarrow \underline{\hspace{1cm}}$$

Examples:

```
\underline
```

The primitive `\underline` is defined in the set `tex`.

The Primitive `\unexpanded`

`\unexpanded` is not implemented yet.

The primitive `\unexpanded` is defined in the set `etex`.

The Primitive `\unhbox`

The formal description of this primitive is the following:

$$\langle unhbox \rangle \rightarrow \unhbox \langle 8\text{-bit number} \rangle$$

Examples:

```
\unhbox42
```

The primitive `\unhbox` is defined in the set `tex`.

The Primitive `\unhcopy`

The formal description of this primitive is the following:

$$\langle unhcopy \rangle \rightarrow \unhcopy \langle 8\text{-bit number} \rangle$$

Examples:

```
\unhcopy42
```

The primitive `\unhcopy` is defined in the set `tex`.

The Primitive `\unkern`

The formal description of this primitive is the following:

$\langle unkern \rangle$
 $\rightarrow \text{\texttt{\textbackslash unkern}}$

Examples:

```
\unkern
```

The primitive `\unkern` is defined in the set `tex`.

The Primitive `\unless`

Copied of the $\epsilon\text{-T}_{\text{E}}\text{X}$ reference.

$\text{T}_{\text{E}}\text{X}$ has, by design, a rather sparse set of conditional primitives: `\ifeof`, `\ifodd`, `\ifvoid`, etc., have no complementary counterparts. Whilst this normally poses no problems since each accepts both a `\then` (implicit) and an `\else` (explicit) part, they fall down when used as the final `\if...` of a `\loop ... \if ... \repeat` construct, since no `\else` is allowed after the final `\if...` `\unless` allows the sense of all Boolean conditionals to be inverted, and thus (for example) `\unless \ifeof` yields true iff end-of-file has not yet been reached.

The formal description of this primitive is the following:

Examples:

```
\unless\if\x\y not ok \fi
```

The primitive `\unless` is defined in the set `etex`.

The Primitive `\unnaturaldir`

`\unnaturaldir` is not implemented yet.

The primitive `\unnaturaldir` is defined in the set `omega`.

The Primitive `\unpenalty`

The formal description of this primitive is the following:

$\langle unpenalty \rangle$
 $\rightarrow \text{\texttt{\textbackslash unpenalty}}$

Examples:

```
\unpenalty
```

The primitive `\unpenalty` is defined in the set `tex`.

The Primitive `\unskip`

The formal description of this primitive is the following:

$$\langle \textit{unskip} \rangle \\ \rightarrow \texttt{\backslash unskip}$$

Examples:

```
\unskip
```

The primitive `\unskip` is defined in the set `tex`.

The Primitive `\unvbox`

The formal description of this primitive is the following:

$$\langle \textit{unvbox} \rangle \\ \rightarrow \texttt{\backslash unvbox} \langle 8\text{-bit number} \rangle$$

Examples:

```
\unvbox42
```

The primitive `\unvbox` is defined in the set `tex`.

The Primitive `\unvcopy`

The formal description of this primitive is the following:

$$\langle \textit{unvcopy} \rangle \\ \rightarrow \texttt{\backslash unvcopy} \langle 8\text{-bit number} \rangle$$

Examples:

```
\unvcopy42
```

The primitive `\unvcopy` is defined in the set `tex`.

The Primitive `\uppercase`

The formal description of this primitive is the following:

$$\langle \textit{uppercase} \rangle \\ \rightarrow \texttt{\backslash uppercase} \langle \dots \rangle$$

Examples:

```
\uppercase ...
```

The primitive `\uppercase` is defined in the set `tex`.

The Primitive `\vadjust`

The formal description of this primitive is the following:

$\langle vadjust \rangle$
 $\rightarrow \text{\vadjust ...}$

Examples:

```
\vadjust{\kern2pt}
```

The primitive `\vadjust` is defined in the set `tex`.

The Primitive `\valign`

The formal description of this primitive is the following:

$\langle span \rangle$
 $\rightarrow \text{\valign}$

Examples:

```
\valign
```

The primitive `\valign` is defined in the set `tex`.

The Primitive `\vbadness`

`\vbadness` is a count register. The primitive `\vbadness` is defined in the set `tex`.

The Primitive `\vbox`

The contents of the toks register `\everyvbox` is inserted at the beginning of the vertical material of the box.

The formal description of this primitive is the following:

$\langle vbox \rangle$
 $\rightarrow \text{\vbox} \langle box\ specification \rangle \{ \langle vertical\ material \rangle \{$
 $\langle box\ specification \rangle$
 \rightarrow
 $\quad | \quad \text{to} \langle rule\ dimension \rangle$
 $\quad | \quad \text{spread} \langle rule\ dimension \rangle$

Examples:

```
\vbox{abc}
```

```
\vbox to 120pt{abc}
```

```
\vbox spread 12pt{abc}
```

The tokens parameter is used in `/vbox`. The tokens contained are inserted at the beginning of the vertical material of the vbox.

The primitive `\vbox` is defined in the set `tex`.

The Primitive `\vcenter`

The formal description of this primitive is the following:

$$\langle vcenter \rangle \rightarrow \backslash \backslash$$

Examples:

```
\backslash
```

The primitive `\vcenter` is defined in the set `tex`.

The Primitive `\vfi`

The formal description of this primitive is the following:

$$\langle vfi \rangle \rightarrow \backslash vfi$$

Examples:

```
\vfi
```

The primitive `\vfi` is defined in the set `omega`.

The Primitive `\vfil`

The formal description of this primitive is the following:

$$\langle vfil \rangle \rightarrow \backslash vfil$$

Examples:

```
\vfil
```

The primitive `\vfil` is defined in the set `tex`.

The Primitive `\vfill`

The formal description of this primitive is the following:

$$\langle vfill \rangle \rightarrow \text{\code{\vfill}}$$

Examples:

```
\vfill
```

The primitive `\vfill` is defined in the set `tex`.

The Primitive `\vfilneg`

The formal description of this primitive is the following:

$$\langle vfilneg \rangle \rightarrow \text{\code{\vfilneg}}$$

Examples:

```
\vfilneg
```

The primitive `\vfilneg` is defined in the set `tex`.

The Primitive `\vfuzz`

`\vfuzz` is a dimen register. The primitive `\vfuzz` is defined in the set `tex`.

The Primitive `\voffset`

`\voffset` is a dimen register. The primitive `\voffset` is defined in the set `tex`.

The Primitive `\vrule`

This primitive produces a vertical rule. This is a rectangular area of specified dimensions. If not overwritten the height and depth are 0pt and the width is 0.4 pt (26214 sp).

The formal description of this primitive is the following:

$$\begin{aligned} \langle vrule \rangle &\rightarrow \text{\code{\vrule}} \langle rule\ specification \rangle \\ \langle rule\ specification \rangle &\rightarrow \langle optional\ spaces \rangle \\ &\quad | \quad \langle rule\ dimension \rangle \langle rule\ specification \rangle \end{aligned}$$

$\langle rule\ dimension \rangle$
 \rightarrow `width` $\langle dimen \rangle$
 $|$ `height` $\langle dimen \rangle$
 $|$ `depth` $\langle dimen \rangle$

The color from the typographic context is taken as foreground color for the rule. The default color is black.

Examples:

```
\vrule
```

```
\vrule height 2pt
```

```
\vrule width 2pt depth 3mm height \dimen4
```

The primitive `\vrule` is defined in the set `tex`.

The Primitive `\vsize`

`\vsize` is a `dimen` register. The primitive `\vsize` is defined in the set `tex`.

The Primitive `\vskip`

The formal description of this primitive is the following:

$\langle vskip \rangle$
 \rightarrow `\vskip` $\langle Glue \rangle$

Examples:

```
\vskip 1em plus 1pt minus 1pt
```

The primitive `\vskip` is defined in the set `tex`.

The Primitive `\vsplit`

The formal description of this primitive is the following:

$\langle vsplit \rangle$
 \rightarrow `\vsplit`

Examples:

```
\vsplit ...
```

The primitive `\vsplit` is defined in the set `tex`.

The Primitive `\vss`

The formal description of this primitive is the following:

$$\langle vss \rangle \rightarrow \backslash vss$$

Examples:

```
\vss
```

The primitive `\vss` is defined in the set `tex`.

The Primitive `\vtop`

The contents of the toks register `\everyvbox` is inserted at the beginning of the vertical material of the box.

The formal description of this primitive is the following:

$$\begin{aligned} \langle vtop \rangle &\rightarrow \backslash vtop \langle box\ specification \rangle \{ \langle vertical\ material \rangle \{ \\ \langle box\ specification \rangle & \\ &\rightarrow \\ &| \quad \text{to} \langle rule\ dimension \rangle \\ &| \quad \text{spread} \langle rule\ dimension \rangle \end{aligned}$$

Examples:

```
\vtop{abc}
```

```
\vtop to 120pt{abc}
```

```
\vtop spread 12pt{abc}
```

The primitive `\vtop` is defined in the set `tex`.

The Primitive `\wd`

The formal description of this primitive is the following:

$$\langle wd \rangle \rightarrow \backslash wd \langle 8\text{-bit}\ number \rangle \langle equals \rangle \langle dimen \rangle$$

Examples:

```
\wd42
```

The primitive `\wd` is defined in the set `tex`.

The Primitive `\widowpenalties`

`\widowpenalties` is not implemented yet.

The primitive `\widowpenalties` is defined in the set `etex`.

The Primitive `\widowpenalty`

`\widowpenalty` is a count register. The primitive `\widowpenalty` is defined in the set `tex`.

The Primitive `\write`

The primitive `\write` is defined in the set `tex`.

The Primitive `\xdef`

The formal description of this primitive is the following:

```

<xdef>
  → <prefix> \xdef <control sequence> <parameter text> { <replacement text> }
<prefix>
  →
  | \global <prefix>
  | \long <prefix>
  | \outer <prefix>

```

Examples:

```
\xdef#1{--#1--}
```

The primitive `\xdef` is defined in the set `tex`.

The Primitive `\xleaders`

The formal description of this primitive is the following:

```

<xleaders>
  → \xleaders ...

```

Examples:

```
\xleaders\hrul\hfill
```

The primitive `\xleaders` is defined in the set `tex`.

4. *The Macro Language of $\epsilon\chi\mathrm{T}_E\mathrm{X}$*

The Primitive `\xspaceskip`

`\xspaceskip` is a skip register. The primitive `\xspaceskip` is defined in the set `tex`.

The Primitive `\year`

`\year` is a count register. The primitive `\year` is defined in the set `tex`.

4.2. Basic Syntactic Entities of $\epsilon\chi\mathrm{T}_E\mathrm{X}$

A. Licenses

A.1. GNU Free Documentation License

Version 1.2, November 2002
Copyright © 2000,2001,2002 Free Software
Foundation, Inc.

51 Franklin St, Fifth Floor, Boston, MA
02110-1301 USA

Everyone is permitted to copy and distribute
verbatim copies of this license document, but
changing it is not allowed.

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary

Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this

definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after

A. Licenses

the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title

of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright ©YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the

"with... Texts." line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

A.2. GNU Library General Public License

Version 2, June 1991

Copyright © 1991 Free Software Foundation, Inc. 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the library GPL. It is numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users.

This license, the Library General Public License, applies to some specially designated Free Software Foundation software, and to any other libraries whose authors decide to use it. You can use it for your libraries, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link a program with the library, you must provide complete object files to the recipients so that they can relink them with the library, after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright the library, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain that everyone understands

that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the library.) The hope is that this will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from

the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

- This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of

A. Licenses

the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:
 - a) The modified work must itself be a software library.
 - b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
 - c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
 - d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful. (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then re-link to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- c) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:
 - a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
 - b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.
8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.
10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy,

distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Index

Symbols

-	20
.extex	11
\	18, 27
&	21
\	28
\\	18

A

\above	18, 28
\abovedisplayshortskip	18, 28
\abovedisplayskip	18, 28
\abovewithdelims	18, 28
\accent	18, 29
\addafterocplist	18, 29
\addbeforeocplist	18, 29
\adjdemerits	18, 29
\advance	18, 30
\afterassignment	18, 30
\aftergroup	18, 30
\atop	18, 31
\atopwithdelims	18, 31

B

\badness	18, 31
\baselineskip	18, 31
batchmode	13, 22
\batchmode	18, 32
\begingroup	18, 32
\beginL	17, 32
\beginR	17, 32
\belowdisplayshortskip	18, 32
\belowdisplayskip	18, 32
\binoppenalty	18, 32
\botmark	18, 33
\botmarks	17, 33
\box	18, 33
\boxmaxdepth	18, 33
\brokenpenalty	18, 33

C

\catcode	18, 34
\char	18, 34
\chardef	18, 34
\cleaders	18, 35
\clearocplists	18, 35
\closein	18, 35
\closeout	18, 35
\clubpenalties	17, 36

\clubpenalty	18, 36
-configuration	21
\copy	18, 36
-copyright	21
\count	18, 36
\countdef	18, 36
\cr	18, 37
\crr	18, 37
\csname	18, 37
CTAN	7
\currentgrouplevel	17, 38
\currentgrouptype	17, 38
\currentifbranch	17, 38
\currentiflevel	17, 38
\currentifttype	17, 38
CVS	8

D

\day	18, 38
\deadcycles	18, 38
-debug	21
\def	18, 39
\defaultthyphenchar	18, 39
\DefaultInputMode	18, 39
\DefaultInputTranslation	18, 39
\DefaultOutputMode	18, 39
\DefaultOutputTranslation	18, 39
\defaultskewchar	18, 40
\delcode	18, 40
\delimiter	18, 40
\delimiterfactor	18, 40
\delimitershortfall	18, 41
\detokenize	17, 41
\dimen	18, 41
\dimendef	18, 41
\dimenexpr	17, 41
\discretionary	18, 41
\displayindent	18, 42
\displaylimits	18, 42
\displaystyle	18, 42
\displaywidowpenalties	17, 42
\displaywidowpenalty	18, 42
\displaywidth	18, 42
\divide	18, 43
\doublehyphendemerits	18, 43
\dp	18, 44
\dump	18, 44

E

\edef	18, 45
\else	18, 45

Index

<code>\emergencystretch</code>	18, 45
<code>\end</code>	18, 20, 45
<code>\endcsname</code>	18, 46
<code>\endgroup</code>	18, 46
<code>\endinput</code>	18, 46
<code>\endL</code>	17, 47
<code>\endlinechar</code>	18, 47
<code>\endR</code>	17, 47
<code>\eqno</code>	18, 47
<code>\errhelp</code>	18, 47
<code>\errmessage</code>	18, 47
<code>\errorcontextlines</code>	18, 48
<code>errorstopmode</code>	13, 22
<code>errorstopmode</code>	18, 48
<code>\escapechar</code>	18, 48
<code>\eTeXrevision</code>	17, 48
<code>\eTeXversion</code>	17, 48
<code>\everycr</code>	18, 48
<code>\everydisplay</code>	18, 48
<code>\everyeof</code>	17, 48
<code>\everyhbox</code>	18, 49
<code>\everyjob</code>	18, 49
<code>\everymath</code>	18, 49
<code>\everypar</code>	18, 49
<code>\everyvbox</code>	18, 49
<code>\exhyphenpenalty</code>	18, 49
<code>\expandafter</code>	18, 49
Explorer	9
<code>\export</code>	17, 50
extex	9–11, 16, 20
extex-jx	16
extex-native	16
ExTeX-setup.jar	8–11
extex.bat	9–11
extex.code	12, 20
extex.config	12, 21
extex.encoding	12
extex.error.handler	12
extex.file	13, 20
extex.fmt	13, 21
extex.fonts	12
extex.halt.on.error	12, 21
extex.ini	13, 22
extex.interaction	13, 22
extex.jobname	13, 22
extex.jobname.master	13
extex.lang	14, 22
extex.nobanner	14
extex.output	14, 22
extex.outputdir	14, 23
extex.outputdir.fallback	14, 23
extex.progname	15, 23
extex.stacktrace.on.internal.error	15
extex.texinputs	11, 12, 15, 23
extex.trace.font.files	15, 21
extex.trace.input.files	15, 21
extex.trace.macros	15, 21
extex.trace.tokenizer	16, 21
extex.typesetter	16

F

<code>\fam</code>	18, 50
<code>\fi</code>	18, 50
<code>\finalhyphendemerits</code>	18, 50
<code>\firstmark</code>	18, 51

<code>\firstmarks</code>	17, 51
<code>\floatingpenalty</code>	18, 51
-fmt	21
<code>\font</code>	18, 52
<code>\fontchardp</code>	17, 52
<code>\fontcharht</code>	17, 52
<code>\fontcharic</code>	17, 53
<code>\fontcharwd</code>	17, 53
<code>\fontdimen</code>	18, 53
<code>\fontname</code>	18, 54
<code>\futurelet</code>	18, 54

G

<code>\gdef</code>	18, 54
<code>\global</code>	18, 55
<code>\globaldefs</code>	18, 55
<code>\glueexpr</code>	17, 55
<code>\glueshrink</code>	17, 55
<code>\glueshrinkorder</code>	17, 56
<code>\gluestretch</code>	17, 56
<code>\gluestretchorder</code>	17, 56

H

<code>\halign</code>	18, 56
-halt-on-error	21
<code>\hangafter</code>	18, 56
<code>\hangindent</code>	18, 56
<code>\hbadness</code>	18, 57
<code>\hbox</code>	18, 57
-help	22
<code>\hfi</code>	18, 57
<code>\hfil</code>	18, 58
<code>\hfill</code>	18, 58
<code>\hfilneg</code>	18, 58
<code>\hfuzz</code>	18, 58
<code>\hoffset</code>	18, 58
<code>\holdinginserts</code>	18, 59
<code>\hrule</code>	18, 59
<code>\hsize</code>	18, 59
<code>\hskip</code>	18, 60
<code>\hss</code>	18, 60
<code>\ht</code>	18, 60
<code>\hyphenation</code>	18, 60
<code>\hyphenchar</code>	18, 61
<code>\hyphenpenalty</code>	18, 61

I

<code>\if</code>	18, 61
<code>\ifcase</code>	18, 61
<code>\ifcat</code>	18, 61
<code>\ifcsname</code>	17, 62
<code>\ifdefined</code>	17, 62
<code>\ifdim</code>	18, 62
<code>\ifeof</code>	18, 62
<code>\iffalse</code>	18, 63
<code>\ifhbox</code>	18, 63
<code>\ifhmode</code>	18, 64
<code>\ifinner</code>	18, 64
<code>\ifmmode</code>	18, 64
<code>\ifnum</code>	18, 65
<code>\ifodd</code>	18, 65
<code>\iftrue</code>	18, 65
<code>\ifvbox</code>	18, 66

`\ifvmode`.....18, 66
`\ifvoid`.....18, 67
`\ifx`.....18, 67
`\ignorespaces`.....18, 67
`\immediate`.....18, 67
`\import`.....17, 68
`\indent`.....18, 68
`-ini`.....22
`initTeX`.....22
`\input`.....18, 68
`\inputlineno`.....18, 68
`\InputMode`.....18, 69
`\InputTranslation`.....18, 69
`\insert`.....18, 69
`\insertpenalties`.....18, 69
`installation script`.....10
`installer`.....8–10
 `building`.....10
 `language`.....9
`-interaction`.....22
`\interactionmode`.....17, 69
`\interlinepenalties`.....17, 69
`\interlinepenalty`.....18, 70

J

`Java`.....7, 9, 23
`\javadef`.....17, 71
`\javaload`.....17, 72
`-job-name`.....22
`\jobname`.....18, 72

K

`\kern`.....18, 73

L

`language`.....9
 `installer`.....9
`-language`.....22
`\language`.....18, 73
`\lastbox`.....18, 73
`\lastkern`.....18, 73
`\lastlinefit`.....17, 73
`\lastnodetype`.....17, 73
`\lastpenalty`.....18, 74
`\lastskip`.....18, 74
`\lccode`.....18, 74
`\leaders`.....18, 74
`\left`.....18, 74
`\lefthyphenmin`.....19, 75
`\leftskip`.....19, 75
`\leqno`.....19, 75
`\let`.....19, 75
`\limits`.....19, 75
`\linepenalty`.....19, 76
`\lineskip`.....19, 76
`\lineskiplimit`.....19, 76
`\localbrokenpenalty`.....18, 76
`\localinterlinepenalty`.....18, 76
`\localleftbox`.....18, 76
`\localrightbox`.....18, 76
`log file`.....25
`\long`.....19, 76
`\looseness`.....19, 77

`\lower`.....19, 77
`\lowercase`.....19, 77
`ls-R`.....7

M

`\mag`.....19, 77
`Mailing list`.....5
`\mark`.....19, 78
`\marks`.....17, 78
`\mathaccent`.....19, 78
`\mathbin`.....19, 78
`\mathchar`.....19, 79
`\mathchardef`.....19, 79
`\mathchoice`.....19, 79
`\mathclose`.....19, 80
`\mathcode`.....19, 80
`\mathdir`.....18, 80
`\mathinner`.....19, 80
`\mathop`.....19, 81
`\mathopen`.....19, 81
`\mathord`.....19, 81
`\mathpunct`.....19, 81
`\mathrel`.....19, 82
`\mathsurround`.....19, 82
`\maxdeadcycles`.....19, 82
`\maxdepth`.....19, 82
`\meaning`.....19, 82
`\medmuskip`.....19, 82
`\message`.....19, 82
`\middle`.....17, 83
`\mkern`.....19, 83
`\month`.....19, 83
`\moveleft`.....19, 83
`\moveright`.....19, 84
`\mskip`.....19, 84
`\muexpr`.....17, 84
`\multiply`.....19, 85
`\muskip`.....19, 85
`\muskipdef`.....19, 85

N

`\namespace`.....17, 86
`\natedef`.....18, 86
`\nativeload`.....18, 86
`\naturaldir`.....18, 86
`\newlinechar`.....19, 86
`nextex`.....17
`\noalign`.....19, 87
`\noboundary`.....19, 87
`\noDefaultInputMode`.....18, 87
`\noDefaultInputTranslation`.....18, 87
`\noDefaultOutputMode`.....18, 87
`\noDefaultOutputTranslation`.....18, 87
`\noexpand`.....19, 88
`\noindent`.....19, 88
`\nolimits`.....19, 88
`\nonscript`.....19, 89
`nonstopmode`.....13, 22
`\nonstopmode`.....19, 89
`\nulldelimiterspace`.....19, 89
`\nullfont`.....19, 89
`\nullocplist`.....18, 89
`\number`.....19, 90
`\numexpr`.....17, 90

O

<code>\ocp</code>	18, 90
<code>\ocplist</code>	18, 90
<code>\odelimiter</code>	18, 90
<code>\omathaccent</code>	18, 90
<code>\omathchar</code>	18, 90
<code>\omathchardef</code>	18, 91
<code>\omathcode</code>	18, 91
<code>\omathdelcode</code>	18, 91
<code>omega</code>	17
<code>\omit</code>	19, 91
<code>\openin</code>	19, 91
<code>\openout</code>	19, 91
<code>\or</code>	19, 91
<code>\oradical</code>	18, 92
<code>\outer</code>	19, 92
<code>-output</code>	22
<code>\output</code>	19, 92
<code>\OutputMode</code>	18, 92
<code>\outputpenalty</code>	19, 92
<code>\OutputTranslation</code>	18, 92
<code>\over</code>	19, 93
<code>\overfullrule</code>	19, 93
<code>\overline</code>	19, 93
<code>\overwithdelims</code>	19, 93

P

<code>\pagedepth</code>	19, 93
<code>\pagedir</code>	18, 93
<code>\pagedirHL</code>	18, 94
<code>\pagedirHR</code>	18, 94
<code>\pagediscarts</code>	17, 94
<code>\pagefilllstretch</code>	19, 94
<code>\pagefillstretch</code>	19, 94
<code>\pagefilstretch</code>	19, 94
<code>\pagegoal</code>	19, 94
<code>\pageshrink</code>	19, 94
<code>\pagestretch</code>	19, 94
<code>\pagetotal</code>	19, 95
<code>\par</code>	19, 95
<code>\parfillskip</code>	19, 95
<code>\parindent</code>	19, 95
<code>\parshape</code>	19, 95
<code>\parshapedimen</code>	17, 96
<code>\parshapeindent</code>	17, 96
<code>\parshapelength</code>	17, 96
<code>\parskip</code>	19, 96
<code>path</code>	9, 10
<code>\patterns</code>	19, 96
<code>\pausing</code>	19, 96
<code>pdfTeX</code>	5
<code>\penalty</code>	19, 97
<code>\popocplist</code>	18, 97
<code>\postdisplaypenalty</code>	19, 97
<code>\predisplaydirection</code>	17, 97
<code>\predisplaypenalty</code>	19, 97
<code>\predisplaysize</code>	19, 97
<code>\pretolerance</code>	19, 97
<code>\prevdepth</code>	19, 98
<code>\prevgraf</code>	19, 98
<code>-prognam</code>	23
<code>\protected</code>	17, 98
<code>\pushocplist</code>	18, 98

R

<code>\radical</code>	19, 99
<code>\raise</code>	19, 99
<code>\read</code>	19, 99
<code>\readline</code>	17, 99
<code>\relax</code>	19, 20, 100
<code>\relax</code>	12
<code>\relpenalty</code>	19, 100
<code>\removebeforeocplist</code>	18, 100
<code>repository</code>	8
<code>\right</code>	19, 100
<code>\righthyphenmin</code>	19, 100
<code>\rightskip</code>	19, 100
<code>\romannumeral</code>	19, 101

S

<code>\savingshyphcodes</code>	17, 101
<code>\savingsdiscarts</code>	17, 101
<code>\scantokens</code>	17, 101
<code>\scriptfont</code>	19, 101
<code>\scriptscriptfont</code>	19, 101
<code>\scriptscriptstyle</code>	19, 102
<code>\scriptspace</code>	19, 102
<code>\scriptstyle</code>	19, 102
<code>scrollmode</code>	13, 22
<code>\scrollmode</code>	19, 102
<code>search</code>	25
<code>\setbox</code>	19, 102
<code>\setlanguage</code>	19, 103
<code>\sfcode</code>	19, 103
<code>\shipout</code>	19, 103
<code>\show</code>	19, 104
<code>\showbox</code>	19, 104
<code>\showboxbreadth</code>	19, 104
<code>\showboxdepth</code>	19, 104
<code>\showgroups</code>	17, 104
<code>\showlists</code>	19, 105
<code>\showthe</code>	19, 105
<code>\showtokens</code>	17, 105
<code>\skewchar</code>	19, 105
<code>\skip</code>	19, 105
<code>\skipdef</code>	19, 106
<code>\spacefactor</code>	19, 106
<code>\spaceskip</code>	19, 106
<code>\span</code>	19, 106
<code>\special</code>	19, 107
<code>\splitbotmark</code>	19, 107
<code>\splitbotmarks</code>	17, 107
<code>\splitdiscarts</code>	17, 107
<code>\splitfirstmark</code>	19, 108
<code>\splitfirstmarks</code>	17, 108
<code>\splitmaxdepth</code>	19, 108
<code>\splittopskip</code>	19, 108
<code>\string</code>	19, 108

T

<code>\tabskip</code>	19, 108
<code>TeX</code>	5, 22, 25
<code>tex</code>	17
<code>-texinputs</code>	23
<code>TeXLive</code>	7
<code>texmf</code>	7, 11
<code>-texmfoutputs</code>	23
<code>-texoutputs</code>	23

`\textdir`.....18, 109
`\textfont`.....19, 109
`\textstyle`.....19, 109
`\TeXeTstate`.....17, 109
`\the`.....19, 109
`\thickmuskip`.....19, 109
`\thinmuskip`.....19, 109
`\time`.....19, 110
`\toks`.....19, 110
`\toksdef`.....19, 110
`\tolerance`.....19, 110
`\topmark`.....19, 110
`\topmarks`.....17, 110
`\topskip`.....19, 110
`\tracingassigns`.....17, 111
`\tracingcommands`.....17, 19, 111
`\tracinggroups`.....17, 111
`\tracingifs`.....17, 111
`\tracinglostchars`.....19, 111
`\tracingmacros`.....19, 111
`\tracingnesting`.....17, 111
`\tracingonline`.....19, 111
`\tracingoutput`.....19, 111
`\tracingpages`.....19, 112
`\tracingparagraphs`.....19, 112
`\tracingrestores`.....19, 112
`\tracingscantokens`.....17, 112
`\tracingstats`.....19, 112

U

`\uccode`.....19, 112
`\uchyph`.....19, 112
`\underline`.....19, 113
`\unexpanded`.....17, 113
`\unhbox`.....19, 113
`\unhcopy`.....19, 113
`Unix`.....15, 23
`\unkern`.....19, 114
`\unless`.....17, 114
`\unnaturaldir`.....18, 114
`\unpenalty`.....19, 114
`\unskip`.....19, 115
`\unvbox`.....19, 115
`\unvcopy`.....19, 115
`\uppercase`.....19, 115

V

`\vadjust`.....19, 116
`\valign`.....19, 116
`\vbadness`.....19, 116
`\vbox`.....19, 117
`\vcenter`.....19, 117
`version`.....23
`-version`.....23
`\vfi`.....18, 117
`\vfil`.....19, 117
`\vfill`.....19, 118
`\vfildneg`.....19, 118
`\vfuzz`.....19, 118
`\voffset`.....19, 118
`\vrule`.....19, 119
`\vsize`.....19, 119
`\vskip`.....19, 119
`\vsplit`.....19, 119

`\vss`.....19, 120
`\vtop`.....19, 120

W

`\wd`.....19, 120
`Web Site`.....5
`\widowpenalties`.....17, 121
`\widowpenalty`.....19, 121
`Windows`.....9, 15, 23
`\write`.....19, 121
`WWW`.....5

X

`\xdef`.....19, 121
`\xleaders`.....19, 121
`\xspaceskip`.....19, 122

Y

`\year`.....19, 122