

Das dateiliste-Package – Liste der verwendeten Dateien im Dokument*

Paul Ebermann^{†‡}

27. Juli 2006

Zusammenfassung

Dieses Paket implementiert einige Befehle, um den Überblick über Versionen und Änderungsdaten von L^AT_EX-Quelltexten zu behalten: Automatische Versionsinfos aus dem CVS, Hauptdatei in der Dateiliste, Inklusion der Dateiliste im Dokument.

Inhaltsverzeichnis

1 Benutzerdokumentation der Befehle	1
1.1 README in Englisch, Deutsch und Esperanto	2
1.2 Automatische Versionsinfos	2
1.3 Hauptdatei in der Dateiliste, Package-Optionen	2
1.4 Dateilisten-Ausgabe	3
1.4.1 Liste der Dateinamen	3
1.5 Abhängigkeiten	4
2 Implementation	4
2.1 Optionen	4
2.2 Aktuelle Versionsnummern in der Dateiliste	5
2.3 Dateiliste	6
2.3.1 Ausgabe der Liste	6
2.3.2 Erstellen der Liste	7
2.3.3 Anpassbare Texte und Übersetzungen	9
2.4 Hauptdatei in die Dateiliste	12
2.5 Schluss	14
3 Liste der Änderungen	14
4 Index	15

*Dieses Dokument gehört zu `dateiliste` v0.3, vom 2006/07/27.

[†]E-Mail: `Paul-Ebermann@gmx.de`

[‡]Rolf Niepraschk (`Rolf.Niepraschk@ptb.de`) hat das Package `printfilelist` geschrieben und mir geschickt, dessen Code bildete die Basis für `\printFileList`. Für den jetzigen Code (insbesondere dessen Fehler) bin ich (Paul) aber selbst verantwortlich.

1 Benutzerdokumentation der Befehle

Die drei Teile *automatische Versionsinfos* (mittels `\ProvideFileInfos`), *Hauptdatei in der Dateiliste* (mittels `\mainFileToList` bzw. der Package-Option) und *Dateilisten-Ausgabe* (mittels `\printFileList`) sind unabhängig voneinander nutzbar, aber die ersten beiden Features werden durch das letzte erst richtig nützlich, und damit das letzte eine vernünftige Ausgabe hat, sind die ersten beiden hilfreich. Daher also alles in einem Package.

1.1 README in Englisch, Deutsch und Esperanto

Eine README-Datei mit einer Kurzübersicht des Paketes auf Englisch, Deutsch und Esperanto kann ebenfalls mit docstrip aus der dtx-Datei generiert werden, die beiliegende Datei `dateiliste.ins` tut dies neben der Generierung der `.sty`-Datei.

1.2 Automatische Versionsinfos

`\ProvideFileInfos` $\{\langle id-string \rangle\}\{\langle kurzbeschreibung \rangle\}$
Ändert die Informationen für die Datei, in der es aufgerufen wurde. $\langle id-string \rangle$ sollte ein String sein, der wie

`$Id: dateiliste.dtx,v 2.1 2006/07/27 10:39:02 epaul Exp $`

(Beispiel für diese Datei) aussieht. Diesen lässt man am besten von seinem CVS produzieren, schreibt also etwas wie

`\ProvideFileInfos{$Id:$}\{Algebraische Geometrie I\}`

in seine Datei und macht dann ein `cvcs commit` – dadurch werden von CVS die passenden Informationen eingefüllt.

$\langle kurzbeschreibung \rangle$ sollte eine kurze Beschreibung der Funktion/des Inhaltes der Datei sein, optimalerweise nur ASCII-Zeichen.

Das Makro sorgt dann dafür, dass in der durch `\listfiles` sowie auch der durch `\printFileList` (siehe Abschnitt 1.4) erzeugten Liste die richtigen Daten (d.h. das Commit-Datum und die RCS/CVS-Versionsnummer) stehen.

1.3 Hauptdatei in der Dateiliste, Package-Optionen

`\mainFileToList` Fügt die Haupt-Datei ($\langle jobname \rangle.tex$) am Anfang der Dateiliste ein, falls sie existiert. Zuvor wird versucht, ein etwaiges Vorkommen dieses Namens in der Liste aus dieser zu entfernen. (Ein solches Vorkommen kann nur dann sein, wenn die Haupt-Datei mit `\input{...}` geladen wurde, anstatt einfach so den Namen auf der Kommandozeile anzugeben, oder wenn die Hauptdatei sich selbst noch einmal einliest.)

`cat12` Die interne Funktionsweise (und damit etwaige Nebenwirkungen) hängt davon ab, ob eine der Package-Optionen `cat12` oder `nocat12` gegeben wurde:

- Mit Option `cat12` wird die interne Liste der geladenen Dateien in Kategorie-12-Zeichen (d.h. other) umgewandelt, um den eventuell dort schon vorhandenen Namen der Hauptdatei zu löschen. Dies kann Nebenwirkungen auf andere Pakete haben, wenn diese ebenfalls diese Liste verarbeiten und dabei auf die Kategorien der Zeichen angewiesen sind.

- Mit Option `nocat12` funktioniert es bei Verwendung von ε -TeX auch ohne die eben geschilderte Umwandlung, bei TeX ist das nicht möglich und es wird einfach so der Dateiname eingefügt – mit dem Effekt, dass er eventuell doppelt auftaucht.
- Ohne diese Optionen wird bei Verwendung von ε -TeX die nebenwirkungsfreie Variante, ohne ε -TeX die Kategorie-12-Variante gewählt.

Details dazu sind im Implementationsteil nachzulesen.

Dieses Makro wird automatisch am Ende des Dokumentes ausgeführt, falls `noaddmain` nicht die Package-Option `noaddmain` gesetzt wurde. Mit `addmain` kann das Vorgabe-Verhalten erzwungen werden.

1.4 Dateilisten-Ausgabe

`\printFileList` [*⟨gliederung⟩*]

Fügt an der aktuellen Stelle eine Liste der im aktuellen Dokument verwendeten Dateien (ohne die Haupt-Datei) ein. *⟨gliederung⟩* ist ein Gliederungsbefehl (wie `\section`, `\chapter` etc., Vorgabewert ist `\section*` (für einen unnummerierten Abschnitt)).

Damit eine Liste ausgegeben wird, muss in der Präambel des Dokumentes ein `\listfiles` auftauchen. (Andernfalls gibt `\printFileList` nur eine Warnung auf der Konsole aus und tut sonst nichts.)

Die Liste selbst ist erst ab dem zweiten L^AT_EX-Lauf im Dokument zu sehen (und enthält ab dem dritten Lauf dann auch die Dateilisten-Datei).

Die Liste wird als 4-spaltige Tabelle gesetzt.

Falls das Paket `babel` vor oder nach diesem Paket geladen wird, sind die Spaltenüberschriften, die Gliederung sowie die Präambel auch übersetzbar – zur Zeit werden von diesem Paket die Sprachoptionen `english`, `german`, `ngerman` und `esperanto` unterstützt.

Ansonsten kann durch Neudefinition der Befehle `\fileListName` (Überschrift), `\fileListPreamble` (einleitender Text, normalerweise mit Fußnote), sowie `\fileNameName`, `\dateName`, `\verName` und `\descriptionName` (Tabellenkopf) der Inhalt der statischen Texte verändert werden. (Auch das wirkt sich erst im folgenden L^AT_EX-Lauf aus.)

Im folgenden mal ein Beispiel aus diesem Dokument, erzeugt mit:

```
\printFileList[\subsubsection]
```

1.4.1 Liste der Dateinamen

Hier die Liste aller Dateien, die während des L^AT_EX-Laufes, welcher dieses Dokument erzeugte, verwendet wurden.¹

Dateiname	Datum	Ver.	Beschreibung
ltxdoc.cls	1999/08/08	v2.0u	Standard LaTeX documentation class
ltxdoc.cfg	2001/06/05	v1.0	teTeX: LaTeX documentation setup

¹genauer: Es ist die Liste aller Dokumente, die einen L^AT_EX-Lauf früher verwendet wurden. Aber nach einigen Läufen sollte sich die Liste stabilisieren.

Dateiname	Datum	Ver.	Beschreibung
article.cls	2004/02/16	v1.4f	Standard LaTeX document class
size10.clo	2004/02/16	v1.4f	Standard LaTeX file (size option)
doc.sty	2004/02/09	v2.1b	Standard LaTeX documentation package (FMi)
multicol.sty	2006/05/18	v1.6g	multicolumn formatting (FMi)
pauldoc.sty	2006/04/05	v0.4a	Pauls Anpassungen fuer doc (PE)
inputenc.sty	2004/02/05	v1.0d	Input encoding file
latin1.def	2004/02/05	v1.0d	Input encoding file
babel.sty	2004/11/20	v3.8d	The Babel package
ngermanb.ldf	2004/02/20	v2.6m	new German support from the babel system
fontenc.sty	—		
t1enc.def	2004/02/22	v1.99f	Standard LaTeX file
dateiliste.sty	2006/07/27	v0.3	Ausgabe der Dateiliste (PE)
rcsinfo.sty	2005/02/20	v1.10	
ltxtable.sty	1995/12/11	v0.2	longtable/tabularx merge (DPC)
tabularx.sty	1999/01/07	v2.07	‘tabularx’ package (DPC)
array.sty	2005/08/23	v2.4b	Tabular extension package (FMi)
longtable.sty	2004/02/01	v4.11	Multi-page Table package (DPC)
enumerate.sty	1999/03/05	v3.00	enumerate extensions (DPC)
dateiliste.dtx	—		
t1cmss.fd	1999/05/25	v2.5h	Standard LaTeX font definitions
t1cmtt.fd	1999/05/25	v2.5h	Standard LaTeX font definitions
omscmr.fd	1999/05/25	v2.5h	Standard LaTeX font definitions
dateiliste.filelist	2006/07/27	—	automatically generated filelist
dateiliste.gls	—		
dateiliste.ind	—		

1.5 Abhängigkeiten

Für die Funktion des Paketes sind die Pakete `rcsinfo` (Jürgen Vollmer) und `ltxtable` – damit auch `tabularx`, `longtable` (alle drei von David Carlisle) und `array` (Frank Mittelbach) – notwendig. `babel` (Johannes Braams) wird, falls ebenfalls geladen, auch genutzt.

Falls das Paket `pauldoc` (von mir) ebenfalls geladen wird, werden einige spezielle Anpassungen getroffen.

Für die korrekte Erkennung, dass der Name der Hauptdatei schon in der Liste der geladenen Dateien auftaucht, ist der primitive ε -TeX-Befehl `\scantokens` notwendig – falls kein ε -TeX verwendet wird, kann eine nicht-Erkennung (und damit am Ende das doppelte Auftauchen der Haupt-Datei) vorkommen.

2 Implementation

```
1 %<*package>
```

2.1 Optionen

Wir merken uns die ausgewählte Option in einem `\if`.

```
2 \newif\if@dateiliste@addMain
```

```

3 \DeclareOption{addmain} {%
4   \@dateiliste@addMaintrue
5 }
6 \DeclareOption{noaddmain} {%
7   \@dateiliste@addMainfalse
8 }

```

Das gleiche gilt für cat12/nocat12.

```

9 \newif\if@dateiliste@catxii \@dateiliste@catxiifalse
10 \newif\if@dateiliste@nocatxii \@dateiliste@nocatxiifalse
11 \DeclareOption{cat12} {%
12   \@dateiliste@catxiitrue
13 }
14 \DeclareOption{nocat12} {%
15   \@dateiliste@nocatxiitrue
16 }

```

Die Standard-Option ist addmain.

```

17 \ExecuteOptions{addmain}
18 \ProcessOptions

```

2.2 Aktuelle Versionsnummern in der Dateiliste

Zunächst wollen wir in der durch `\listfiles` provozierten Ausgabe automatisch sinnvolle Infos haben. Dafür laden wir das Paket `rcsinfo`. Wir verwenden die Parameter `nofancy`, weil sonst die Fußzeile umgestellt wird, und `notoday`, weil sonst das aktuelle Datum umgestellt wird.

```

19 \RequirePackage[nofancy, notoday]{rcsinfo}

```

```

\ProvideFileInfos {<id-string>}{<kurzbeschreibung>}

```

```

20 \newcommand*{\ProvideFileInfos}[2] {%

```

Zunächst lassen wir `\rcsInfo` den `<id-string>` analysieren. Dies definiert (unter anderem) die Makros `\rcsInfoFile` (der Dateiname), `\rcsInfoDate` (Datum, im YYYY/MM/DD-Format) und `\rcsInfoRevision` (die Versionsnummer).

Das Leerzeichen nach dem #1 ist notwendig, damit `\rcsInfo` erkennt, wo der `<id-string>` aufhört - in der Definition steht da nämlich ein Leerzeichen am Ende der Parameterliste.

```

21   \rcsInfo #1 %

```

Dann rufen wir `\ProvidesFile` aus dem L^AT_EX-Kernel auf.

```

22   \ProvidesFile%

```

Als erster Parameter wird der Dateiname übergeben, der von `\rcsInfo` ermittelt wurde. Mittels `\expandafter\@firstofone` entfernen wir dabei noch das von `\rcsInfo` (zumindest in meiner Version) eingebaute Leerzeichen am Anfang (welches ja einen anderen Namen ergibt und damit verhindern würde, dass die Information der richtigen Datei zugeschrieben wird).

```

23   {\expandafter\@firstofone\rcsInfoFile}%

```

Dann das, wofür wir das ganze eigentlich machen: Das Datum, ein Leerzeichen, dann die Versionsnummer (mit einem `v` davor). Schließlich hängen wir noch `<kurzbeschreibung>` an.

```

24   [\rcsInfoDate\space v\rcsInfoRevision\space #2]%

```

`\ProvidesFile` definiert jetzt ein Makro (`\ver@{dateiname}`) mit diesem Text als Inhalt, welches später von `\dofilelist` (und unserem `\writefilelist`) verwendet wird.

25 }

2.3 Dateiliste

Da die Liste ziemlich lang (länger als eine Seite) werden kann, verwende ich `longtable` statt der eingebauten (oder der von `array` verbesserten) `tabular`-Umgebung. Und damit ich in der letzten Spalte nicht die Breite fest einstellen muss, sondern einfach die restliche Breite (abhängig von Seitenbreite und der Breite der anderen Spalten, welche ja abhängig vom Inhalt ist) nehmen kann, lade ich `ltxtable`, welches `longtable` mit `tabularx` kreuzt (und beide Pakete auch lädt).

26 `\RequirePackage{ltxtable}`

`\dateiliste@preInclude`
`\dateiliste@postInclude`

Diese beiden Macros werden vor bzw. nach dem Laden (und setzen) der Dateiliste aufgerufen. Sie sorgen dafür, dass ' innerhalb der Liste nicht mehr in den Verbatim-Mode schaltet, wie das von `pauldoc` eingestellt wird. Deswegen werden sie auch nur dann so definiert, wenn `pauldoc` geladen wurde. (Und weil `\@ifpackageloaded` nur in der Präambel erlaubt ist, müssen wir die beiden Befehle schon zu Beginn des Dokumentes definieren, anstatt einfach die Abfrage dann zu machen, wenn es gebraucht wird.)

```
27 \AtBeginDocument{%
28   \@ifpackageloaded{pauldoc}{%
29     \newcommand*{\dateiliste@preInclude}{\DeleteShortVerb{'}}%
30     \newcommand*{\dateiliste@postInclude}{\MakeShortVerb{'}}%
31   }{%
32     \newcommand*{\dateiliste@preInclude}{\relax}%
33     \newcommand*{\dateiliste@postInclude}{\relax}%
34   }%
35 }
```

Die beiden Makros kann man sich auch selbst umdefinieren, falls andere Pakete Inkompatibilitäten ergeben.

2.3.1 Ausgabe der Liste

`\printFileList` [*gliederung*]

Der Vorgabewert für *gliederung* ist `\section*`, also ein unnummerierter Abschnitt.

36 `\newcommand*{\printFileList}[1][\section*]{\printFileList`

Zunächst überprüfen wir, ob `\listfiles` in der Präambel gegeben wurde. Dies zeigt sich darin, dass das Kommando `\dofilelist` definiert ist. Andernfalls gibt es eine Warnung, und wir machen nichts.

```
37   \@ifundefined{@dofilelist}
38   {%
39     \PackageWarning{dateiliste}
40     {
41       \protect\printFileList\space works only if
42       \protect\listfiles\space is given in the preamble.
43     }
```

```

44 }%
45 {%                               else (\@ifundefined{dofilelist})

```

Andernfalls beginnen wir einen neuen Abschnitt (oder ein Kapitel oder was auch immer mit *gliederung*) festgelegt wurde), mit Namen `\fileListName` und einem Label, falls man mal von wo anders darauf verweisen möchte. Danach kommt etwas beschreibender Text in `\fileListPreamble`.

```

46 #1{\fileListName}\label{sec:filelist}%
47 \fileListPreamble

```

In der Datei `\jobname.filelist` befindet sich nach dem ersten L^AT_EX-Lauf der Inhalt der Tabelle (siehe unten). Wir überprüfen zunächst, ob die Datei schon existiert.

```

48 \IfFileExists{\jobname.filelist}{%

```

`\dateiliste@addtofilelist` Falls ja, dann definieren wir zunächst `\@addtofilelist` um, da `\LTxtable` die Datei `\jobname.filelist` mehrfach einliest, wir aber nur einen Eintrag in der Dateiliste haben wollen. Wir verwenden nicht einfach `\gobble`, um in dem Fall, dass durch das Setzen der Datei weitere Dateien (Schriften etc.) geladen werden, diese doch aufzunehmen. (Wir vergleichen also den Dateinamen mit dem unserer Dateinamens-Datei, und rufen im Fall der Nichtübereinstimmung das Original-`\@addtofilelist` auf.)

```

49 \let \dateiliste@addtofilelist = \@addtofilelist
50 \def\@addtofilelist####1{%
51 \edef\dateiliste@tempa{####1}%
52 \edef\dateiliste@tempb{\jobname.filelist}\relax%
53 \ifx\dateiliste@tempa\dateiliste@tempb
54 \relax
55 \else
56 \dateiliste@addtofilelist{####1}
57 \fi
58 }%

```

`\dateiliste@preInclude` schaltet ' als verbatim-Char ab (und das Makro `\dateiliste@postInclude` schaltet es nachher wieder an), falls `pauldoc` geladen wurde (ansonsten tun sie nichts, falls nicht von jemand anders neudefiniert). Die Datei selbst wird mittels `\LTxtable` geladen.

```

59 \dateiliste@preInclude
60 \LTxtable{\linewidth}{\jobname.filelist}%
61 \dateiliste@postInclude

```

Danach stellen wir `\@addtofilelist` wieder her und fügen unsere Dateilisten-Datei auch hinzu.

```

62 \let \@addtofilelist = \dateiliste@addtofilelist
63 \@addtofilelist{\jobname.filelist}%
64 }

```

Falls `\jobname.filelist` nicht vorhanden war, geben wir einen Hinweistext aus, dass man L^AT_EX noch einmal laufen lassen soll.

```

65 {%
66 \PackageWarning{dateiliste}{
67 Run LaTeX again to include the File list.
68 }%
69 }%

```

2.3.2 Erstellen der Liste

Jetzt noch ein paar Befehle, um die Listen-Datei zu generieren ... (Wir sind immer noch innerhalb von `\printFileList`, das alles passiert also nur, wenn dieser Befehl aufgerufen wird.)

Am Ende des Dokumentes – d.h., wenn die Dateiliste vollständig gesammelt wurde – schreiben wir sie – mit den passenden Formatierungsanweisungen – in eine Datei. (Das ganze in einer Gruppe, damit nichts kaputtgeht, und temporäre Makros nachher wieder freigegeben werden.)

```
70      \AtEndDocument{%
71      \begingroup
72      \@writefilelist
73      \endgroup
74      }%
```

`\@writefilelist` Eine Variante von `\@dofilelist`, die den Inhalt – als Tabellenzeilen – in die Datei `\jobname.filelist` schreibt.

```
75      \newcommand*{\@writefilelist}{% \@writefilelist
76      \newwrite\dateiliste@file
77      \immediate\openout\dateiliste@file = \jobname.filelist
```

Zunächst schreiben wir eine `\ProvidesFile`-Anweisung mit dem aktuellen Datum in die `.filelist`-Datei. (Das hat den Effekt, dass diese Datei selbst auch in der Liste erscheint.)

```
78      \edef\dateiliste@today{%
79      \the\year/\two@digits{\the\month}/\two@digits{\the\day}}%
80      \immediate\write\dateiliste@file{%
81      \string\ProvidesFile{\jobname.filelist}%
82      [\dateiliste@today\space --- automatically %
83      generated filelist]%
84      }%
```

Die eigentliche Liste wird in einer `longtable` gesetzt. Diese soll drei linksbündig gesetzte Spalten (1) und dann eine mit einem Absatz (X – mittels `ltxtable` aus `tabularx` importiert), welche den restlichen Platz ausfüllt, enthalten. Damit die letzte Spalte linksbündig (statt Blocksatz) wird, verwenden wir `>\raggedright\arraybackslash` als Modifikator ²

```
85      \immediate\write\dateiliste@file{%
86      \string\LTleft=0pt%
87      \string\LTRight=0pt%
88      \string\begin{longtable}{l l l >\string\raggedright\string\arraybackslash}X}%■
```

Die Überschrift – aus übersetzbaren Textteilen, siehe unten, bestehend – wiederholt sich auf jeder Seite (deswegen `\endhead` anstatt `\\`).

```
89      \string\textbf{\fileNameName} &
90      \string\textbf{\dateName} &
91      \string\textbf{\verName} &
92      \string\textbf{\descriptionName}
93      \string\endhead%
94      }%
```

²wie im L^AT_EX-Begleiter, zweite Auflage, Beispiel 5-3-2 vorgeschlagen.

Jetzt kommt die Schleife mit den einzelnen Dateien. Das ist zum Großteil abgekupfert von `\@dofilelist` aus dem L^AT_EX-Kernel (`lTfiles.dtx`), welches die Liste zum Terminal ausgibt.

```
95      \@for\@currname:=\@filelist\do{%   \@for
```

Zunächst bestimmen wir den genauen Dateinamen – d.h. wir hängen, falls nötig, ein `.tex` an. Außerdem finden wir den zugehörigen Versions-String heraus.

```
96      \filename@parse\@currname
97      \edef\dateiliste@filename{%
98          \filename@base.%
99          \ifx\filename@ext\relax tex\else\filename@ext\fi}%
100     \expandafter\let\expandafter\dateiliste@fileversion
101     \csname ver@\dateiliste@filename\endcsname
```

Jetzt schreiben wir, durch `&` getrennt, die einzelnen Felder raus. Zunächst der Dateiname, ...

```
102     \immediate\write\dateiliste@file{%
103         \dateiliste@filename\space& %
```

... dann entweder ein „—“ (falls kein Versions-String gegeben wurde), ...

```
104         \ifx\dateiliste@fileversion\relax
105         ---
106         \else
```

... oder der Versionsstring selbst, an den ersten beiden Leerzeichen durch `&` getrennt. Dafür verfüttern wir das expandierte `\dateiliste@fileversion` an `\dateiliste@parse@ver`. (Für den Fall, dass da nicht genug Leerzeichen drin sind, sind am Ende noch ein paar `{}` mit Leerzeichen dazwischen – die werden am Ende ja nicht ausgegeben.)

```
107         \expandafter\dateiliste@parse@ver
108         \dateiliste@fileversion{} {} {} \relax
109         \fi
```

Und jetzt noch ein `\\`, um die Tabellenzeile zu beenden.

```
110         \string\\}%
111     }%   \@for
```

Nach der Schleife beenden wir die Tabelle und schließen dann die Datei wieder.

```
112     \immediate\write\dateiliste@file{\string\end{longtable}}
113     \immediate\closeout\dateiliste@file
114     }%
```

`\dateiliste@parse@ver` Dieses Makro nimmt zwei durch Leerzeichen getrennte Parameter, und gibt sie, mit zusätzlichen `&`, wieder zurück.

```
115     \def\dateiliste@parse@ver##1 ##2 {##1 & ##2 & }%
```

Damit ist der else-Teil und auch das ganze Makro `\printFileList` zu Ende.

```
116     }%
117 }%
```

2.3.3 Anpassbare Texte und Übersetzungen

`\fileListPreamble` Einige Namen für übersetzbare Texte – standardmäßig auf Englisch.

```
\fileListName 118 \newcommand*\fileListPreamble{
\fileNameName 119 Here is the list of all files used during the run of \LaTeX{}
\dateName
\verName
\descriptionName
```

```

120   which produced this document. \footnote{More precisely, it is
121       the list of files used one \LaTeX-run before the one which
122       produced this document, but after some runs the list should
123       stabilize.}
124 }
125 \newcommand*\fileListName{List of Files}
126 \newcommand*\fileNameName{file name}
127 \newcommand*\dateName{release date}
128 \newcommand*\verName{version}
129 \newcommand*\descriptionName{description}

```

`\dateiliste@babel` Hier noch gleich ein paar Übersetzungen. Wir definieren hier ein einmal-Makro, welches für mehrere Sprachen³ zum jeweiligen Initialisierungsmakro Neudefinitionen dieser fünf Befehle hinzufügt.

```

130 \newcommand*\dateiliste@babel{%
    Zunächst Englisch - das sollte das gleiche wie die Standard-Einstellungen sein.
131     \addto{\extrasenglish}{%
132         \renewcommand*\fileListPreamble{%
133             Here is the list of all files used during the run of \LaTeX{}
134             which produced this document.\footnote{More precisely, it is
135                 the list of files used one \LaTeX-run before the one which
136                 produced this document, but after some runs the list
137                 should stabilize.}
138         }
139         \renewcommand*\fileListName{List of Files}%
140         \renewcommand*\fileNameName{file name}
141         \renewcommand*\dateName{release date}
142         \renewcommand*\verName{ver.}
143         \renewcommand*\descriptionName{description}
144     }%

```

Deutsch mit neuer Rechtschreibung.

```

145     \addto{\extrasgerman}{%
146         \renewcommand*\fileListPreamble{%
147             Hier die Liste aller Dateien, die während des \LaTeX-Laufes,
148             welcher dieses Dokument erzeugte, verwendet wurden.
149             \footnote{genauer: Es ist die Liste aller Dokumente, die
150                 einen \LaTeX-Lauf früher verwendet wurden. Aber nach
151                 einigen Läufen sollte sich die Liste stabilisieren.}
152         }
153         \renewcommand*\fileListName{Liste der Dateinamen}%
154         \renewcommand*\fileNameName{Dateiname}
155         \renewcommand*\dateName{Datum}
156         \renewcommand*\verName{Ver.}
157         \renewcommand*\descriptionName{Beschreibung}
158     }%

```

Deutsch mit alter Rechtschreibung: ist das gleiche (hier tauchen keine Fälle mit Änderungen auf.)

```

159     \addto{\extrasngerman}{%
160         \renewcommand*\fileListPreamble{%

```

³Genauer: genau für die Sprachen, welche ich soweit beherrsche, dass ich diese Texte übersetzen konnte.

```

161      Hier die Liste aller Dateien, die während des \LaTeX-Laufes,
162      welcher dieses Dokument erzeugte, verwendet wurden.
163      \footnote{genauer: Es ist die Liste aller Dokumente, die
164      einen \LaTeX-Lauf früher verwendet wurden. Aber nach
165      einigen Läufen sollte sich die Liste stabilisieren.}
166  }
167  \renewcommand*\fileListName{Liste der Dateinamen}%
168  \renewcommand*\fileNameName{Dateiname}
169  \renewcommand*\dateName{Datum}
170  \renewcommand*\verName{Ver.}
171  \renewcommand*\descriptionName{Beschreibung}
172  }%

```

Für die Verwender der Internationalen Sprache (siehe www.esperanto.de):

```

173  \addto{\extrasesperanto}{%
174    \renewcommand*\fileListPreamble{%
175      Jen listo de \^ciuj dosieroj, kiuj estis uzitaj dum
176      la \LaTeX-rulo, kiu produktis tiun \^ci dokumenton.
177      \footnote{Pli precize: estas la listo de dosieroj uzitaj
178      unu rulon anta\u{u} tiu, kiu produktis tiun \^ci
179      dokumenton. Sed kutime post kelkaj ruloj la listo
180      devus stabili\^gi.}
181    }
182    \renewcommand*\fileListName{Listo de dosieroj}%
183    \renewcommand*\fileNameName{dosiernomo}
184    \renewcommand*\dateName{dato}
185    \renewcommand*\verName{versio}
186    \renewcommand*\descriptionName{priskribo}
187  }%

```

Am Ende der Ausführung von `\dateiliste@babel` vernichtet der Befehl sich selbst. Das spart etwas Speicher, und sorgt dafür, dass er nicht versehentlich mehrfach ausgeführt wird (auch wenn das wohl nicht schädlich wäre).

```

188  \let \dateiliste@babel = \relax%
189  }%

```

Wir untersuchen jetzt, ob `babel` schon geladen wurde. Diese Fallunterscheidung ist notwendig, weil der Code von `\dateiliste@babel` zwar das Paket benötigt (also nach ihm ausgeführt werden sollte), aber nicht einfach direkt mit `\AtBeginDocument` ans Ende geschoben werden sollte, da er (falls `babel` schon vor diesem Paket geladen wurde) dort nach dem `babel`-Code (der die Sprache auswählt) kommen würde, und damit mehr nichts bewirkt.

Falls `babel` jetzt schon geladen wurde, ...

```

190 \@ifpackageloaded{babel}
191 {%
...informieren wir es sofort über die neuen Namen, die beim Sprachwechsel bitte
angepasst werden sollten.
192  \dateiliste@babel%
193  }%

```

Ansonsten verschieben wir das zum Beginn des Dokumentes (und machen das auch dann nur, wenn `babel` inzwischen geladen wurde – ansonsten ist das ganze ja überflüssig, und `\addto` gibt es auch nicht, also können wir dann `\dateiliste@babel` vernichten).

```

194 {%
195   \AtBeginDocument{%
196     \ifpackageloaded{babel}{%
197       \dateiliste@babel%
198     }{%
199       \let \dateiliste@babel = \relax
200     }%
201   }%
202 }%

```

2.4 Hauptdatei in die Dateiliste

`\mainFileToList` Zunächst sehen wir nach, ob es eine Datei mit Namen `\jobname.tex` gibt.

```

203 \newcommand*{\mainFileToList}{% \mainFileToList
204   \IfFileExists{\jobname.tex} {%
205     \begingroup

```

Falls ja, dann ist das höchstwahrscheinlich die Haupt-Datei des Dokumentes, und taucht wahrscheinlich – nämlich, wenn sie auf der Kommandozeile oder mit `\input \jobname.tex` anstatt `\input{\jobname.tex}` geladen wurde – nicht in der Dateiliste auf.

Das Problem an der Erkennung des letzten Falles (L^AT_EX-`\input{}`) ist, dass `\jobname` die Zeichen (auch die Buchstaben) in Kategorie 12 (*other*) liefert anstatt in der natürlichen Kategorie (d.h. Buchstaben in 11 = *letter*) (wie die Dateien, deren Name irgendwo im Quelltext auftaucht und dann in `\@filelist` landet).

Ich habe drei Möglichkeiten gefunden, damit umzugehen:

- (1) Wir wandeln `\@filelist` komplett in Kategorie-12-Zeichen um. (Den Tipp, wie das geht, habe ich zufällig beim Durchstöbern der UK-T_EX-FAQ gefunden: <http://www.tex.ac.uk/cgi-bin/texfaq2html?label=compjobnam>)
- (2) Wir wandeln `\jobname` (bzw. die Buchstaben darin) in ihre „richtige“ Kategorie (11 für Buchstaben) um.
- (3) Wir ignorieren das Problem und leben damit, dass eventuell der Dateiname doppelt auftaucht.

(2) funktioniert leider nur bei Verwendung von ε -T_EX, (1) hat den Nachteil, dass anschließend `\@filelist` komplett aus Kategorie-12-Zeichen besteht, was eventuell zu Problemen mit anderen Paketen führt, welche ebenfalls diese Liste verarbeiten und auf deren Catcodes angewiesen sind.

Daher haben ich zwei Paket-Optionen hinzugefügt, welche eine entsprechende Auswahl ermöglichen:

Standardvorgehen: Falls ε -T_EX verwendet wurde, nimm (2), für T_EX nimm (1).

Mit Option `nocat12`: Falls ε -T_EX verwendet wurde, nimm (2), sonst (3).

Mit Option `cat12`: Nimm immer (1).

(Vor Version 0.2 gab es nur das Verhalten, welches jetzt `nocat12` entspricht.)

`\dateiliste@catxii@transform` Hier die Implementation für (1). Wir wandeln auch das `.tex` in Kategorie 12 um, mittels `\meaning` (mit `\strip@prefix` entfernen wir etwas Text, der vor der Makrodefinition steht)

```

206     \newcommand*{\dateiliste@catxii@transform}
207     {
208         \edef\dateiliste@mainfile{\jobname.tex}%
209         \edef\dateiliste@mainfile{%
210             \expandafter\strip@prefix\meaning\dateiliste@mainfile
211         }%
212     }
213     \edef\@filelist{\expandafter\strip@prefix\meaning\@filelist}%

```

Jetzt das gleiche für \@filelist.

\dateiliste@scantoken@tr Hier die Implementation für (2) mit \scantokens: In ε -TeX gibt es dagegen den \scantokens-Befehl, welcher es ermöglicht, im Speicher von TeX vorliegende Token neu aus einer Pseudo-Datei einzulesen. Wenn er definiert ist (nur dann wird dieses Makro aufgerufen), rufen wir ihn hier auf – mit einigen \expandafter, um nur den \jobname vor der \scantokens-Ausführung zu expandieren, und nicht die Token \edef\dateiliste@mainfile{ und .tex} drumherum. \scantokens bekommt also die Zeichen

\makeatletter\edef\dateiliste@mainfile{\jobname}.tex\makeatother zu lesen.

Das \edef wird dann also mit Kategorie-11-Buchstaben (also „richtigen“) im Dateinamen ausgeführt. (Das \makeatletter und \makeatother ist notwendig, um das @ auch als Buchstabe zuzulassen und somit \dateiliste@mainfile als einzelnen Makronamen anzusehen. Zu dem Zeitpunkt, zu dem \scantokens ausgeführt wird, sind ja die Dokumenten-Catcodes in Kraft, nicht die einer Package-Datei.)

```

214     \newcommand*{\dateiliste@scantoken@tr}
215     {%
216         \scantokens
217         \expandafter{%
218             \expandafter\makeatletter
219             \expandafter\edef
220             \expandafter\dateiliste@mainfile
221             \expandafter{%
222                 \jobname
223                 .tex}%
224             \makeatother
225         }%
226     }%

```

Hier jetzt die große Fallunterscheidung:

```

227     \if@dateiliste@catxii
228         \dateiliste@catxii@transform
229     \else
230         \@ifundefined{scantokens}
231         {%
232             \if@dateiliste@nocatxii

```

Das ist die Implementation für (3): wir merken uns den Dateinamen einfach so. Das geht wahrscheinlich schief, wenn nicht noch irgend ein anderes Makropaket eingreift, und wir können \jobname.tex nicht aus der Liste entfernen.

```

233         \edef\dateiliste@mainfile{\jobname.tex}%
234     \else

```

```

235         \dateiliste@catxii@transform
236         \fi
237     }
238     {%
239         \dateiliste@scantoken@tr
240     }%
241     \fi

```

Jetzt haben wir in `\dateiliste@mainfile` den Namen der Hauptdatei, auch (je nach Optionen und ε -TEX-Verfügbarkeit) `@filelist` vorbereitet.

Wir können nun mit `\@removeelement` (aus dem L^AT_EX-Kernel) das Vorkommen von `<jobname>.tex` entfernen (falls der Name dort vorhanden ist – wahrscheinlich nicht).

```

242     \expandtwoargs\@removeelement{\dateiliste@mainfile}%
243     \@filelist\@filelist

```

Anschließend fügen wir den Dateinamen an den *Anfang* der Liste an.

```

244     \xdef\@filelist{%
245         \dateiliste@mainfile,\@filelist
246     }%
247     \endgroup

```

Falls `<jobname>.tex` nicht existiert, ist dies sicher nicht die Hauptdatei. Dann haben wir es entweder mit einer `.dtx`-Datei zu tun (die sowieso durch das doppelte Einlesen noch einmal auftaucht), oder irgendeinen anderen Fall, den ich nicht vorhersehen kann. Also machen wir dann nichts.

```

248     }{%
249         \relax
250     }%
251 }%

```

Am Ende des Dokumentes (aber noch vor dem Aufruf von `\@writefilelist`, der von `\printFileList` hinzugefügt wird) rufen wir, sofern die passende Option gesetzt war, das eben definierte Makro auf.

```

252 \if@dateiliste@addMain
253   \AtEndDocument{\mainFileToList}
254 \fi

```

2.5 Schluss

```

255 \endinput %
256 \</package>

```

3 Liste der Änderungen

v0.0	Optionen	addmain	und	
Allgemein: Erste Fassung	1	noaddmain	hinzugefügt.	3
v0.1		\dateiliste@addtofilelist:	Neu	6
\@addtofilelist: neu: Umdefiniti-		\dateiliste@parse@ver:	Neu	8
on.	6	\dateiliste@postInclude:	Neu	5
Allgemein: ltxtable verwendet. . . .	5	\dateiliste@preInclude:	Neu	5
rcsinfo nun mit notoday-		\dateName:	Neu	8
Parameter.	4	\descriptionName:	Neu	8

\fileListPreamble: Neu	8	nocat12 hinzugefügt.	4
\fileNameName: Neu	8	README-Dateien werden jetzt	
\mainFileToList: Neu	11	auch aus der .dtx-Datei gene-	
\printFileList: Fast komplett		riert.	1
neue Implementation, entspre-		\mainFileToList: Mögliche Erken-	
chend auch anderes Ergebnis. .	5	nung des Dateinamens jetzt	
\verName: Neu	8	auch ohne ε -TeX	11
v0.1a			
Allgemein: Kleine Änderungen der		v0.3	
Dokumentation.	1	\@writefilelist: jetzt auch die	
v0.2		letzte Spalte linksbündig – das	
Allgemein: Optionen cat12 und		sieht besser aus.	7

4 Index

Schräggedruckte Nummern verweisen auf die Seite, auf der der Eintrag beschrieben ist, unterstrichene Nummern zeigen auf die Zeilennummer der Definition, sonstige Zahlen auf die Zeilennummer einer Verwendung.

Symbols			
\'	29, 30	\dateiliste@catxii@transform . . .	
\@addtofilelist	<u>49</u> , 62, 63	<u>206</u> , 228, 235
\@dateiliste@addMainfalse	7	\dateiliste@file	
\@dateiliste@addMaintrue	4	76, 77, 80, 85, 102, 112, 113
\@dateiliste@catxiifalse	9	\dateiliste@filename	97, 101, 103
\@dateiliste@catxiitru	12	\dateiliste@fileversion	100, 104, 108
\@dateiliste@nocatxiifalse	10	\dateiliste@mainfile	
\@dateiliste@nocatxiitru	15	208–210, 220, 233, 242, 245
\@expandtwoargs	242	\dateiliste@parse@ver	107, <u>115</u>
\@filelist	95, 212, 243–245	\dateiliste@postInclude	<u>27</u> , 61
\@firstofone	23	\dateiliste@preInclude	<u>27</u> , 59
\@for	95, 111	\dateiliste@scantoken@tr	<u>214</u> , 239
\@ifpackageloaded	28, 190, 196	\dateiliste@tempa	51, 53
\@ifundefined	37, 45, 230	\dateiliste@tempb	52, 53
\@removeelement	242	\dateiliste@today	78, 82
\@writefilelist	72, <u>75</u>	\dateName	3, 90, <u>118</u>
\\	110	\DeleteShortVerb	29
\^	175, 176, 178, 180	\descriptionName	3, 92, <u>118</u>
A		E	
\addmain	2	\endhead	93
\addto	131, 145, 159, 173	\ExecuteOptions	17
\arraybackslash	88	\extrasenglish	131
\AtBeginDocument	27, 195	\extrasesperanto	173
\AtEndDocument	70, 253	\extrasgerman	145
		\extrasngerman	159
C		F	
\cat12	2	\fileListName	3, 46, <u>118</u>
\closeout	113	\fileListPreamble	3, 47, <u>118</u>
D		\filename@base	98
\dateiliste@addtofilelist	<u>49</u> , 62	\filename@ext	99
\dateiliste@babel	<u>130</u> , 192, 197, 199	\filename@parse	96
		\fileNameName	3, 89, <u>118</u>

I		O	
\if@dateiliste@addMain	2, 252	\openout	77
\if@dateiliste@catxii	9, 227	P	
\if@dateiliste@nocatxii	10, 232	\PackageWarning	39, 66
J		\printFileList	2, <u>36</u>
\jobname	48,	\ProcessOptions	18
52, 60, 63, 77, 81, 204, 208, 222, 233		\ProvideFileInfos	1, <u>20</u>
L		\ProvidesFile	22, 81
\label	46	R	
\linewidth	60	\raggedright	88
\listfiles	42	\rcsInfo	21
\LTleft	86	\rcsInfoDate	24
\LTright	87	\rcsInfoFile	23
\LTXtable	60	\rcsInfoRevision	24
M		\RequirePackage	19, 26
\mainFileToList	2, <u>203</u> , 253	S	
\MakeShortVerb	30	\scantokens	216
\meaning	210, 212	\section	36
N		\strip@prefix	210, 212
\noaddmain	2	V	
\nocat12	2	\verName	3, 91, <u>118</u>