Christian Rank

NICE Version 2

Dokumentation

18/09/90

Abstract

NICE ist ein Makropaket für plainTEX, das grundlegende und auch weiterführende Makros zum einfachen Setzen von Sachtexten bereitstellt. Dabei wird die weitestgehende Kompatibilität zu plainTEX gewahrt, d. h. auch wer NICE benutzt, kann wie gewohnt auch die Befehle von plainTEX weiterverwenden. Dieser Text ist eine vollständige Beschreibung der Handhabung von NICE für die NICE-Versionen 2.0 und höher.

LIZENZBESTIMMUNGEN
FÜR DIE BENUTZUNG UND WEITERGABE VON NICE

Weder das NICE-Makropaket noch diese Anleitung dazu sind Public Domain. Jedoch darf jedermann NICE benutzen.

NICE darf unter folgenden Bedingungen weitergegeben werden:

1. Es werden sowohl das Makropaket NICE.TEX als auch diese Anleitung – entweder in gedruckter Form oder als DVI-Datei – weitergegeben.
2. Es dürfen keine Veränderungen am Makropaket oder an der Anleitung vorgenommen worden sein.
3. Dem Empfänger müssen diese Lizenzbedingungen zur Kenntnis gebracht werden.
4. Die Weitergabe erfolgt kostenlos. Geringfügige Gebühren für Porto und/oder Datenträger sind hiervon ausgenommen.
5. Die Verbreitung darf ausdrücklich auch über Softwarearchive, Mailboxen oder ähnliche Einrichtungen erfolgen, soweit diese Lizenzbestimmungen eingehalten werden.

Der Autor übernimmt keine Gewähr für die Fehlerfreiheit und Zuverlässigkeit von NICE. Jegliche Haftung für Schäden irgendwelcher Art, die durch die Verwendung von NICE entstanden sind, ist ausgeschlossen.

Programmname:	**NICE.TEX**
Version:	**2.0**
Release-Datum:	**17/09/90**

Inhalt

Vorwort

Da ist sie nun, die lang erwartete Dokumentation von NICE, die auch gleichzeitig einen Upgrade des NICE-Paketes auf die Version 2.0 mit sich bringt. Ich hätte dieses Vorwort auch mit "Warum NICE?" überschreiben können, denn im folgenden will ich etwas genauer ausführen, warum das NICE-Paket überhaupt entstanden ist.

Es war Frühjahr 1987, als ich das erste Mal in meinem Leben von TEX, dem tollen Textsatzsystem hörte. Zuerst nur sehr theoretisch, aber dann erfuhr ich, daß bei uns an der Uni ein Band mit dem TEX-System 'rumlag, man brauchte es nur noch auf dem Rechner zu installieren. Das war relativ schnell getan, und bald konnte ich mich auch praktisch von der Leistungsfähigkeit von TEX überzeugen. Bald war das TEXbook studiert, und, so ausgerüstet, wollte ich dann einen längeren Text (sinnvollerweise eine Anleitung für die Benutzung von TEX am Unirechner) schreiben.

Dabei machte ich allerdings die Erfahrung, die wohl jeder macht, der das erste Mal mit plainTEX einen längeren Text schreiben will, der auch noch gut gestaltet sein soll: Im Prinzip ist plainTEX ein riesiger Werkzeugkasten, der viele, viele Kleinteile enthält, die wirklich keine Wünsche offenlassen. Nur: Die größeren Trümmer, die man eben für das Schreiben von Texten immer wieder braucht, sind nicht fertig vorhanden. Man muß sie sich mühsam aus den Kleinteilen zusammenbasteln. Das tat ich dann auch am Anfang. Natürlich stellte sich dann heraus, daß man die Basteleien auch in den meisten anderen Texten sehr gut verwenden konnte.

Damit entstand dann die Urversion von NICE. Sie hatte 77 Zeilen und unterstützte im wesentlichen die Einteilung eines Textes in numerierte Abschnitte bis hin zu Unterunterabschnitten; es gab eine numerierte Fußnotenverwaltung und gestaffelte Listen. Diese Urversion wurde dann nach und nach erweitert; Mitte 1988 gab es zu der dann aktuellen Version 1.3 sogar eine Kurzanleitung. Nun, heute gibt es schließlich die Version 2.0 mit einem Umfang von etwa 550 Zeilen, und was diese alles kann, lesen Sie dann in den folgenden Abschnitten.

Nun werden wahrscheinlich einige Leser einen gewichtigen Einwand gegen die Behauptung bringen, daß man mehr oder weniger gezwungen sei, sich beim Arbeiten mit TEX die zweckmäßigen Makros selbst zusammenzubasteln. Es gibt ja LATEX, das einem dieses Aufwands enthebt. Gegen die Verwendung von LATEX sprechen für mich aber einige Gründe. Unbestreitbar ist, daß man mit LATEX relativ schnell sehr beeindruckende Ergebnisse beim Textsatz zustandebringt. Es entfällt auch die Notwendigkeit, sich in die "Niederungen" von plainTEX zu begeben.

Aber: So schön LATEX zum Setzen von Texten ist, wenn man nur die Möglichkeiten verwendet, die LATEX selbst anbietet – wenn man etwas machen will, was in LATEX nicht

standardmäßig oder auch überhaupt nicht vorgesehen ist, wird's schwierig. Nicht nur, daß man sich dann sehr gut mit TEX selbst auskennen muß (sprich: das TEXbook kein Buch mit sieben Siegeln mehr sein sollte), in LATEX sind auch viele Makros von plainTEX gar nicht mehr anwendbar, weil sie entweder gar nicht mehr funktionieren oder nicht mit den LATEX-Makros zusammenarbeiten. Auf den Punkt gebracht, fehlt LATEX die Kompatibilität mit plainTEX, und das ist ein sehr schweres Manko für jemanden, der frei in der Gestaltung seiner Texte sein will.[1]

Also: Wer auf plainTEX schwört, wird NICE sicherlich als komfortable Erleichterung beim Formatieren von Texten schätzen lernen. Denn NICE ist mit plainTEX kompatibel, d. h. (fast)[2] alle Makros und sonstigen Befehle, die plainTEX kennt, können bei der Benutzung von NICE weiter eingesetzt werden.

Zum vollständigen Verständnis dieser Dokumentation sollten Sie grundsätzliche Kenntnisse über die Arbeitsweise von TEX haben, so daß Sie zumindest in der Lage sind, bei Bedarf einige Punkte im TEXbook oder einem äquivalenten Buch über TEX nachzuschlagen.

NICE ist nicht zuletzt auch durch die Anregungen oder Verbesserungsvorschläge der NICE-Benutzer weiterentwickelt worden. Sollten also auch Sie interessante Anregungen, Verbesserungsvorschläge oder ähnliches haben, wenden Sie sich mich über meine untenstehende Adresse. Das gilt natürlich auch, wenn Sie noch einen Fehler in diesem NICE-Paket gefunden haben sollten.[3]

Viel Spaß beim Arbeiten mit NICE wünscht Ihnen

Christian Rank
Bräugasse 13/225
D-8390 Passau

E-Mail: (Bitte nur in Deutschland benutzen)
rank@unipas.fmi.uni-passau.de

[1] Um Mißverständnissen vorzubeugen: Damit soll keineswegs LATEX madig gemacht werden. Viele Leute, die an plainTEX wahrscheinlich verzweifelt wären, verwenden LATEX und sind damit hochzufrieden, und sie können es auch weiterhin sein.

[2] Auf dieses "fast" wird in einem eigenen Abschnitt genauer eingegangen. Keine Angst, Sie kaufen also nicht die Katze im Sack.

[3] In diesem Fall geben Sie bitte bei der Beschreibung des Fehlers auch die NICE-Versionsnummer, die TEX-Versionsnummer und Ihr Rechnersystem an und fügen Sie möglichst einen kurzen (!) TEX-Text bei, bei dem der Fehler aufgetreten ist.

1. Textgestaltungsmöglichkeiten

Die wesentlichen Eigenschaften von NICE werden in diesem und dem nächsten Abschnitt beschrieben. In diesem Abschnitt geht es um die Textgestaltungsmöglichkeiten, die NICE bietet, soweit sie das Gesamtlayout des Textes betreffen.

Das Gesamtlayout eines Sachtextes ist meistens ähnlich dem folgenden:

- **Titelseite** mit Autor- und Titelangabe sowie weiteren Kurzangaben. Eventuell enthält die Titelseite noch eine Kurzzusammenfassung des Inhaltes.
- **Inhaltsverzeichnis**, in das die Überschriften der Abschnitte (und eventuell auch weiterer Untergliederungen) aufgenommen werden.
- **Abschnitte**, die mit einem durchnumerierten Titel überschrieben werden. Jeder Abschnitt kann auch noch weiter in **Unterabschnitte** und weiter in **Unterunterabschnitte** untergliedert sein.
- **Anhang**, z. B. ein Literaturverzeichnis. Besteht der Anhang aus mehreren Abschnitten, so sind diese mit einem durchbuchstabierten Titel versehen.
- Jede Seite (außer die erste) enthält **Kopf-** und **Fußzeilen**, in die Informationen über die Seitennummer und den Titel des auf der Seite beginnenden oder fortgesetzten Abschnitts aufgenommen werden können.

Auf genau diese Gestaltungsmöglichkeiten werden wir im Rest dieses Abschnittes eingehen. Alle anderen Möglichkeiten, die NICE sonst noch anbietet, werden dann im folgenden Abschnitt behandelt.

1.1. Die Titelseite

Man sagt, der erste Eindruck sei der Entscheidende, ganz egal, ob es dabei um neue Bekanntschaften oder das Lesen eines neuen Textes geht. Beim Lesen eines Textes entsteht der erste Eindruck beim Betrachten der ersten Seite des Textes. Bei längeren Texten wird dies meist eine Titelseite sein, die darum ansprechend, übersichtlich und informativ gestaltet sein sollte.

Wichtige Informationen, die die Titelseite enthalten sollte, sind:

– Der Name des Autors bzw. der Autoren, denn der Leser will schließlich wissen, von wem das vorliegende Werk stammt.
– Noch wichtiger ist der Titel des Werkes, damit der Leser schon mal weiß, worum's geht.
– Gegebenenfalls Informationen über die Institution, an der die Arbeit entstanden ist. Das Datum der Fertigstellung kann auch ganz interessant sein. Dies waren zwei Beispiele für weitere Kurzinformationen, die sich auch noch ganz gut auf der Titelseite unterbringen lassen.
– Um das Interesse des Lesers weiter zu steigern, kann man auch noch einen "Abstract" auf der Titelseite unterbringen. Das ist eine kurze Inhaltsangabe des Textes – nicht

so ausführlich wie eine Inhaltsangabe, wie man sie in der Schule schreiben lernt, aber doch etwas detaillierter als eine reine Umformulierung des Titels.

So, damit hätten wir für die Titelseite alles beieinander, was wir brauchen. Mit NICE läßt sich dann die Titelseite ganz einfach zustandebringen. Am Anfang der Seite[4] steht

`\titlepage`

Man hat dann folgende Makros zur Auswahl:

`\title{`*titel*`}`

setzt den Titel des Dokuments zentriert mit dem Zeichensatz `\documenttitlefont` (siehe den Abschnitt über die Konfigurierung von NICE). Der Titel muß nicht unbedingt in eine Zeile passen; er wird automatisch umgebrochen, wenn er zu lang für eine Zeile ist. Um dem Titel den ihm gebührenden Platz einzuräumen, wird nach dem Titel ein `\medskip` ausgeführt. Mit

`\author{`*(auch längerer) text*`}`

kann sich der Autor den ihm gebührenden Tribut erweisen. Es ist hier auch für den Fall vorgesorgt, daß ganze Autorenkollektive NICE benutzen, denn man kann auch einen längeren Text[5] angeben, der durch Absätze getrennt ist. Es wird wieder alles zentriert gesetzt, und, weil Bescheidenheit bekanntlich eine Zier ist, in normalem `\rm`-Zeichensatz[6]. Zwischen Absätzen, die im `\author`-Text vorkommen, wird kein Abstand gelassen (das ist wohl ganz sinnvoll).

`\release{`*einzeiliger text*`}`

ist dazu da, die zusätzlichen Kurzinformationen auf der Titelseite unterzubringen. Sie werden im Fettdruck als zentrierte Zeile gesetzt, dürfen also jeweils nicht länger als eine Druckzeile sein. Selbstverständlich kann man dieses Makro (wie übrigens die anderen auch) wiederholt verwenden.

`\abstract`

leitet zum letzten Teil der Titelseite über, dem "Abstract" (so er denn vorkommen soll). Der Abstract ist immer das letzte, was auf der Titelseite gesetzt wird. Wenn dieses Makro verwendet wird, wird alles, was danach bis zum Ende der Titelseite kommt, in den Abstract aufgenommen. Dieser sieht dann so aus: Als Überschrift steht ein zentriertes "Abstract" (was sonst); der folgende Text wird dann mit Randausgleich 3 cm von linken und 3 cm von rechten Rand entfernt gesetzt.

`\endtitlepage`

[4] Es sollte wirklich vorher kein Text auf dieser Seite stehen, sonst beschwert sich TEX über unter- oder überfüllte `\vbox`en.

[5] hier sinnvoll: die Namen der Autoren

[6] und wer sich lieber nach dem zweiten Teil des Verses über die Bescheidenheit richtet, kann das selbstverständlich ändern, indem er am Anfang des *text*es einen anderen Zeichensatz selektiert.

beendet schließlich die Titelseite. Sie wird dann vertikal in die Seitenmitte zentriert, und zwar vermittels `\vfil` am Anfang und am Schluß, so daß es sich bei Vorhandensein eines Abstracts eigentlich immer empfiehlt, vor Beginn des Abstracts ebenfalls ein `\vfil` in den Text einzufügen.

Nun hierzu ein kleines Beispiel:

```
\titlepage
\author {\bf Christian Rank}
\medskip
\title {NICE Version 2}
\centerline {Dokumentation}
\bigskip
\release {../../..}
\vfil
\abstract
NICE ist ein Makropaket f\"ur plain\TeX,
[...]
f\"ur die NICE-Versionen~2.0 und h\"oher.
\endtitlepage
```

Was NICE bzw. TEX daraus macht, sehen Sie auf der ersten Seite dieser Anleitung[7].

1.2. Abschnitte, Unterabschnitte, Unterunterabschnitte

Das generelle Konzept von Abschnitten und ihren Untergliederungen wurde ja schon eingangs eingeführt. Bei einem Abschnitt (bzw. einer Untergliederung davon) handelt es sich um einen betitelten und meist auch numerierten Textteil. NICE übernimmt die Verwaltung der Abschnitte (und der Untergliederungen davon, aber das habe ich jetzt zum letzten Mal dazugesagt) und stellt dazu die folgenden Makros bereit:

`\plainsection{`*überschrift*`}`

setzt *überschrift* als nicht numerierten Titel eines neuen Abschnitts. Bis auf die fehlende Numerierung ist die Funktion die gleiche wie bei

`\section{`*überschrift*`}`

das *überschrift* ebenfalls als Titel eines neuen Abschnittes erzeugt, allerdings eine laufende Nummer, gefolgt von einem Punkt[8], davorschreibt. Die Numerierung der Abschnitte beginnt bei 1. Der Zähler `\sectionno` enthält immer die Nummer des gerade aktuellen

[7] Die Punkte beim `\release` sind natürlich durch ein aktuelles Datum ersetzt.

[8] Es werden immer wieder heiße Diskussionen darüber geführt, ob nun die Numerierung eines Abschnittes mit einem Punkt abschließen sollte oder nicht. Ich meine, sie sollte, aber es gibt auch andere Meinungen (etwa die meisten LATEX-Benutzer haben eine andere, denn LATEX schreibt grundsätzlich keinen Punkt hinter die Abschnittsnumerierung, aber das nur nebenbei). Keine Sorge, es wird im Abschnitt 4 gesagt, wie man den Numerierungsstil ändert.

Abschnitts, d. h. wer gerne mit Abschnitt 0 beginnen möchte, muß irgendwo am Anfang des TEX-Dokuments die Anweisung `\sectionno=-1` geben.

Das `\section`-Makro bewirkt ziemlich viele Dinge, die wir uns jetzt haarklein ansehen: Erstmal wird der vorhergehende Abschnitt mit einem `\bigskip` abgeschlossen (wobei ein eventuell vorhergehender `\vskip` entfernt wird), dann die `\sectionno` um 1 erhöht und die `\subsectionno`[9] auf 0 zurückgesetzt. Dann wird der Titel des Abschnitts zunächst mit einem `\parskip` eingeleitet und dann mit dem Zeichensatz `\sectiontitlefont` gesetzt. Paßt die Überschrift nicht in eine Zeile, so wird sie automatisch umgebrochen und sogar in der nächsten Zeile hinter die Numerierung eingerückt. Ist `\contentslevel` > 0, so wird die Abschnittsüberschrift samt Numerierung und der Seitennummer, auf der die Überschrift steht, ins Inhaltsverzeichnis eingetragen (siehe Abschnitt 1.3), falls ein solches erzeugt wird. Die Überschrift und die Numerierung werden während des Übersetzens des Dokuments übrigens, wenn sie ins Inhaltsverzeichnis aufgenommen werden, am Bildschirm angezeigt. Schließlich, wird, falls `\marklevel` > 0, die Überschrift (samt Numerierung) für die Anzeige in der Kopfzeile markiert (siehe Abschnitt 1.4). Es wird übrigens automatisch verhindert, daß nach Beginn eines neuen Abschnitts (oder einer Untergliederung) ein Seitenumbruch stattfindet.

So, damit hätten wir das Schwierigste schon hinter uns, denn die Funktion von

`\subsection{`*überschrift*`}`

ist nicht recht viel anders: Der vorhergehende Text wird mit einem `\smallskip` abgeschlossen (dieser ersetzt einen eventuell vorhergehenden `\vskip`), die `\subsectionno` um 1 erhöht, die `\subsubsectionno` auf 0 zurückgesetzt. Die Überschrift wird nach einem `\parskip` samt Numerierung (in der Form `\sectionno`, "Punkt", `\subsectionno`, "Punkt") im Zeichensatz `\subsectiontitlefont` gesetzt. Sie wird ins Inhaltsverzeichnis bei `\contentslevel` > 1 aufgenommen (und dann auch am Bildschirm angezeigt). Entsprechend erfolgt eine Markierung für die Anzeile in der Kopfzeile bei `\marklevel` > 1.

Entsprechend gibt's für Unterunterabschnitte

`\subsubsection{`*überschrift*`}`

Hier wird der vorhergehende Text mit `\par` abgeschlossen, die `\subsubsectionno` um 1 erhöht. Die Überschrift wird nach einem `\parskip` mit kompletter Numerierung im Zeichensatz `\subsubsectiontitlefont` gesetzt. Für `\contentslevel` bzw. `\marklevel` > 2 passiert entsprechend das gleiche wie schon oben zur Genüge beschrieben.

Nun dazu ein kurzes Beispiel: Die Überschriften von diesem Abschnitt und diesem Unterabschnitt wurden mit

[9] Was wird das wohl sein? Wer's nicht errät: Weiter unten folgt die Auflösung.

```
\section {Textgestaltungsm\"oglichkeiten}
\subsection {Abschnitte, Unterabschnitte, Unterunterabschnitte}
```

erzeugt.

Abschließend sei noch kurz die Gestaltung eines Anhangs erwähnt. Es genügt, am Beginn eines Anhangs den Befehl

```
\appendix
```

zu geben. Die einzige Wirkung dieses Befehls ist, die Numerierung der Abschnitte (nicht der weiteren Untergliederungen) auf alphanumerische Form umzustellen, d. h. der nächste Abschnitt bekommt dann die Nummer "A.", der übernächste die Nummer "B.". Ansonsten ändert sich überhaupt nichts am `\section`-Makro. Ich glaube, hier können wir auf ein Beispiel verzichten.

1.3. Das Inhaltsverzeichnis

Für einen umfangreicheren Text ist ein Inhaltsverzeichnis unumgänglich. Abgesehen davon, daß es schon den Aufbau des Textes offenlegt, ist es auch später zum Nachschlagen unentbehrlich. In das Inhaltsverzeichnis werden üblicherweise die Nummern und Überschriften der Abschnitte (gegebenenfalls auch der weiteren Untergliederungen) aufgenommen, zusammen mit der Nummer der Seite, auf der der betreffende Abschnitt beginnt.

Die Erzeugung des Inhaltsverzeichnisses geht mit NICE mehr oder weniger vollautomatisch. An der Stelle, wo das Inhaltsverzeichnis im Text stehen soll, geben Sie den Befehl

```
\makecontents
```

und das Wesentliche ist erledigt. Ein Inhaltsverzeichnis wird immer in einer Art "Zweiphasenmethode" erstellt. Wenn `\makecontents` im Text auftaucht, wird die Datei, in der das Inhaltsverzeichnis bereits formatiert steht,[10] eingelesen und in den Text eingefügt. Während der Übersetzung des restlichen Dokuments wird das aktuelle Inhaltsverzeichnis auf der gleichen Datei erzeugt.

Dies führt dazu, daß Sie Ihren Text zweimal hintereinander übersetzen müssen, wenn Sie garantiert ein richtiges Inhaltsverzeichnis haben wollen. Dies wird jedoch im allgemeinen nur bei der endgültigen Fertigstellung des Textes erforderlich sein. Wichtig ist, daß Sie sich wirklich sonst um nichts bezüglich des Inhalts des Inhaltsverzeichnisses kümmern müssen.

Der genaue Umfang und das genaue Aussehen des Inhaltsverzeichnisses wird durch eine Anzahl von weiteren Makros festgelegt, die wir uns jetzt ansehen werden. Bereits kennengelernt haben wir

```
\contentslevel=zahl
```

[10] Die Datei heißt genauso wie der Text der übersetzt wird, nur statt der Extension TEX hat die Datei die Extension TOC.

Damit wird festgelegt, ob nur Abschnitte (*zahl*=1), Abschnitte und Unterabschnitte (*zahl*=2) oder Abschnitte, Unterabschnitte und Unterunterabschnitte (*zahl*=3) ins Verzeichnis aufgenommen werden. Standardmäßig wird `\contentslevel=1` gesetzt.

Jede Zeile des Inhaltsverzeichnisses besteht aus der (numerierten) Überschrift eines Abschnittes (bzw. einer Untergliederung), die linksbündig gesetzt wird, und der rechtsbündig gesetzten Seitennummer, die angibt, auf welcher Seite der betreffende Abschnitt beginnt. Der Platz zwischen Überschrift und Seitennummer wird üblicherweise freigelassen; manchmal sieht man aber auch, daß dieser Raum mit Punkten aufgefüllt ist. Das geht auch mit NICE; man kann nämlich allgemein mit

`\contfill={`*füller*`}`

irgendeinen *füller* angeben, der dann in den ansonsten leeren Raum eingefügt wird. Geeignete *füller* kann man sich mit dem plainTEX-Kommando `\leaders` erzeugen, oder man verwendet die bereits vorhandenen `\dotfill` oder `\hrulefill`. Standardmäßig ist übrigens – wen wundert's – `\contfill={\hfil}`.

Abschnitte, Unterabschnitte und Unterunterabschnitte werden im Inhaltsverzeichnis verschieden weit eingerückt. Dadurch wird die hierarchische Gliederung des Textes besser erkennbar. Standardmäßig werden Abschnitte gar nicht eingerückt, Unterabschnitte so weit, daß der Beginn der Nummer des Unterabschnitts mit dem Beginn der Überschrift des Abschnittes horzontal übereinstimmt; entsprechendes gilt für die Einrückung von Unterunterabschnitten in Bezug auf Unterabschnitte. Für die Einrückung können Sie selbst eigene Werte wählen, und zwar mit

`\scindent=`*dimension*

für Abschnitte,

`\sscindent=`*dimension*

für Unterabschnitte, und

`\ssscindent=`*dimension*

für Unterunterabschnitte.

Schließlich ist es auch noch möglich, den im Inhaltsverzeichnis verwendeten Schriftstil individuell verschieden für Abschnitte, Unterabschnitte und Unterunterabschnitte zu gestalten. Normalerweise wird alles im normalen `\rm`-Stil gesetzt. Modifikationen erreichen Sie mit

`\let\scstyle=\`*stilmakroname*

für Abschnitte, entsprechend für Unterabschnitte (`\sscstyle`) und Unterunterabschnitte (`\ssscstyle`). *stilmakroname* ist der Name eines Makros, das sinnvollerweise einen anderen Zeichensatz einstellt; wenn Sie also die Einträge für Abschnitte fett geschrieben haben wollen, schreiben Sie `\let\scstyle=\bf`. Es ist auch möglich, den Schriftstil als neues

Makro zu definieren, also mit

`\def\scstyle{...}`

Dies empfiehlt sich etwa dann, wenn man mehrere Befehle zur Erreichung des gewünschten Effekts benötigt und daher `\let` nicht verwenden kann.

Als letzte Möglichkeit zur übersichtlichen Gliederung des Inhaltsverzeichnisses gibt es noch

`\contskip=`*dimension*

Damit geben Sie an, wiviel zusätzlicher Leerraum vor dem Titel eines Abschnitts im Verzeichnis gelassen werden soll. Standardmäßig wird kein zusätzlicher Leerraum gelassen. Beachten Sie bitte, daß `\contskip` gesetzt werden muß, *bevor* `\makecontents` ausgeführt wird, da das erzeugte Inhaltsverzeichnis den `\contskip`-Parameter benutzt.

In dieser Anleitung wurde die Seite mit dem Inhaltsverzeichnis im wesentlichen folgendermaßen erzeugt:

```
\newpage
\contskip=12pt
\mark{Inhalt}
\vbox to \vsize{\baselineskip=18pt minus 6pt
\vfil
\plainsection {Inhalt}
\smallskip
\makecontents
\vfil}
\contentslevel=2
\newpage
```

1.4. Kopf- und Fußzeilen

Die Einrichtung der Kopf- und Fußzeilen kennen Sie ja schon von plainTEX. Standardmäßig ist ja in plainTEX und auch in NICE die Kopfzeile (`\headline`) leer und die Fußzeile (`\footline`) enthält die zentrierte Seitennummer. Das ist zwar sehr einfach, aber nicht sehr schön. NICE bietet ein ansprechenderes Format: Mit

`\article`

wird die Fußzeile leer gemacht (Sie können sie natürlich dann wieder beliebig setzen), und die Kopfzeile enthält linksbündig Nummer und Titel des auf der Seite beginnenden Abschnitts (bzw. Unterabschnitts oder Unterunterabschnitts, je nach Wert des Parameters `\marklevel`) in Kursivschrift und rechtsbündig in Fettschrift die Seitennummer. Beginnt auf der betreffenden Seite kein neuer Abschnitt (oder ggf. eine Untergliederung davon), werden Nummer und Titel des aktuellen Abschnitts (bzw. der Untergliederung) verwendet. Auf der Seite 1 (genauer: Auf allen Seiten mit Nummer ≤ 1) ist die Kopfzeile jedoch leer, um eine eventuell vorhandene Titelseite voll zur Geltung kommen zu lassen.

Fast genaso wie `\article` arbeitet

`\book`

lediglich mit dem Unterschied, daß auf Seiten mit *gerader* Nummer die Seitennummer linksbündig und Abschnittstitel rechtsbündig gesetzt werden. Damit hat man bei zweiseitig bedrucktem, gebundenen Papier die Seitennummern zur besseren Orientierung immer auf dem äußeren Rand des Papiers.

In dieser Anleitung wurden die Kopfzeilen mit
`\book \marklevel=2`
erzeugt.

2. Spezielle Anwendungen

Unter die Kategorie "Spezielle Anwendungen" fallen im Prinzip alle Gestaltungsmöglichkeiten von NICE, die noch nicht im vorhergehenden Abschnitt behandelt wurden, weil sie weniger im Zusammenhang mit dem Gesamtlayout eines Dokuments stehen. Im einzelnen sind dies:

- **Titelzeilen**, ohne daß eine extra Titelseite verwendet wird.
- Verschiedene **Schriftgrößen** im laufenden Text oder für größere Textabschnitte.
- **Fußnoten**, die eine Einfügung von Bemerkungen oder Zitaten ermöglichen, ohne daß der normale Textfluß unterbrochen wird.
- **Gestaffelte Listen** zur gefälligen Gestaltung von Listen mit Einträgen der Form *Schlüsselwort/-zeichen Erläuterung.*
- Der **Verbatim-Modus**, mit dem Textabschnitte Zeichen für Zeichen so gesetzt werden können, wie sie in der Textdatei stehen.
- **Querverweise** in andere Textabschnitte oder auf andere Seiten.
- Verwendung von gewissen **Mathematiksymbolen** im laufenden Text.
- Ansprechendes **Setzen von Programmcode** als Alternative zur üblichen Verwendung des Verbatim-Modus für diesen Zweck.
- Automatische **Numerierung von Abbildungen**.
- **Einrahmen von Texten**, die besonders hervorgehoben werden sollen.

2.1. Titelzeilen

Nicht immer will man gleich eine ganze Titelseite verbraten, um einen Leser für den Text zu interessieren. Bei kürzeren Texten tut's dann auch eine oder mehrere Titelzeilen. Diese erzeugen Sie mit[11]

`\title{`*titel*`}`

[11] Dieses Makro hat nichts mit dem gleichnamigen Makro zu tun, das Sie innerhalb einer Titelseite zur Verfügung haben.

Der *titel* wird mit `\documenttitlefont` zentriert in einer oder ggf. auch mehreren Zeilen gesetzt. Der *titel* darf auch Absätze enthalten.

2.2. Schriftgrößen

Verschiedene Schriftgrößen können Sie schon in plainTEX verwenden, indem Sie sich einen Zeichensatz in der gewünschten Größe mit dem `\font`-Kommando definieren. Das Problem ist hierbei nur: Wird zwischendurch auf eine andere Schriftart umgeschaltet (etwa mit `\it`), so haben Sie für diese Schriftart die standardmäßig geltende Schriftgröße. Dies gilt auch für die Umschaltung auf den Mathematikmodus.

Die Lösung dieses Problems liefern NICE-Makros, die *alle* Schriftarten auf eine bestimmte Größe umstellen, also `\rm`, `\it`, `\sl`, `\bf`, `\tt`, `\oldstyle`, `\textfont0`, `\textfont1` und `\textfont2`. `\scriptfont0`, `\scriptfont1`, `\scriptfont2`, `\scriptscriptfont0`, `\scriptscriptfont1` und `\scriptscriptfont2` werden der neuen Größe angepaßt. Zusätzlich wird `\baselineskip` der gewählten Größe angepaßt. Bei allen Größen wird der mit `\magnification` gesetzte globale Vergrößerungsfaktor berücksichtigt.

Makro	Größe Text	Größe Subskript	Größe Subsubskript	Zeilenabstand
`\ninept`	9 pt	7 pt	5 pt	11 pt
`\eightpt`	8 pt	6 pt	5 pt	10 pt
`\sevenpt`	7 pt	6 pt	5 pt	9 pt
`\sixpt`	6 pt	5 pt	5 pt	8 pt
`\fivept`	5 pt	5 pt	5 pt	7 pt

Nachdem mit einem der obigen Makros eine neue Schriftgröße gewählt wurde, wird automatisch die Schriftart `\rm` gewählt. Es empfiehlt sich, zeitweise Änderungen der Schriftgröße lokal innerhalb einer Gruppe vorzunehmen, da Sie sonst nicht mehr auf die Standardgröße zurückschalten können.

Betrachten Sie den folgenden kurzen Beispieltext:

```
Die Makros zur \"Anderung der Schriftgr\"o\ss{}e \"andern auch die
Gr\"o\ss{}en der
Schriftarten, etwa {\bf Fettdruck}, {\sl Schr\"agschrift}, {\it
Kursivschrift} und {\tt Schreibmaschinenschrift}. Auch der Mathematikmodus
wird in der ge\"anderten Gr\"o\ss{}e gesetzt:
$$ \xi = \sum_{j \in \cal J} x_{i_j} $$
```

In der Abbildung 1 sind die Wirkungen der Größenmakros auf diesen Text gezeigt. Man sieht hier auch deutlich, daß bei den ganz kleinen Schriftgrößen gewisse Schriftarten nicht in der erforderlichen Größe gesetzt werden können, weil TEX sie so klein gar nicht anbietet. Hier wird dann der kleinste verfügbare Zeichensatz in der betreffenden Schriftart genommen.

Standard-Schriftgröße 10 pt:

Die Makros zur Änderung der Schriftgröße ändern auch die Größen der Schriftarten, etwa **Fettdruck**, *Schrägschrift*, *Kursivschrift* und `Schreibmaschinenschrift`. Auch der Mathematikmodus wird in der geänderten Größe gesetzt:

$$\xi = \sum_{j \in \mathcal{J}} x_{i_j}$$

Schriftgröße 9 pt:

Die Makros zur Änderung der Schriftgröße ändern auch die Größen der Schriftarten, etwa **Fettdruck**, *Schrägschrift*, *Kursivschrift* und `Schreibmaschinenschrift`. Auch der Mathematikmodus wird in der geänderten Größe gesetzt:

$$\xi = \sum_{j \in \mathcal{J}} x_{i_j}$$

Schriftgröße 8 pt:

Die Makros zur Änderung der Schriftgröße ändern auch die Größen der Schriftarten, etwa **Fettdruck**, *Schrägschrift*, *Kursivschrift* und `Schreibmaschinenschrift`. Auch der Mathematikmodus wird in der geänderten Größe gesetzt:

$$\xi = \sum_{j \in \mathcal{J}} x_{i_j}$$

Schriftgröße 7 pt:

Die Makros zur Änderung der Schriftgröße ändern auch die Größen der Schriftarten, etwa **Fettdruck**, *Schrägschrift*, *Kursivschrift* und `Schreibmaschinenschrift`. Auch der Mathematikmodus wird in der geänderten Größe gesetzt:

$$\xi = \sum_{j \in \mathcal{J}} x_{i_j}$$

Schriftgröße 6 pt:

Die Makros zur Änderung der Schriftgröße ändern auch die Größen der Schriftarten, etwa **Fettdruck**, *Schrägschrift*, *Kursivschrift* und `Schreibmaschinenschrift`. Auch der Mathematikmodus wird in der geänderten Größe gesetzt:

$$\xi = \sum_{j \in \mathcal{J}} x_{i_j}$$

Schriftgröße 5 pt:

Die Makros zur Änderung der Schriftgröße ändern auch die Größen der Schriftarten, etwa **Fettdruck**, *Schrägschrift*, *Kursivschrift* und `Schreibmaschinenschrift`. Auch der Mathematikmodus wird in der geänderten Größe gesetzt:

$$\xi = \sum_{j \in \mathcal{J}} x_{i_j}$$

Fig. 1: Die wählbaren Schriftgrößen

2.3. Fußnoten

Fußnoten gibt's schon in plainTEX, und an dem grundlegenden Konzept wird auch in NICE nichts geändert. Grundsätzlich eingeführt wurde eine automatische Durchnumerierung, aufgehoben wurde die Abhängigkeit der Fußnoten-Einrückung von `\parindent`.

Die Einrückung wird jetzt durch den Wert von `\footnoteindent` (standardmäßig 20pt) kontrolliert.

`\footnote{`*text*`}`

erzeugt eine numerierte Fußnote, die den angegebenen *text* enthält.[12] Die Numerierung beginnt wie üblich bei 1 und wird nach jeder Fußnote um 1 erhöht. Im Unterschied zu plainTEX wird der Text der Fußnote, auch wenn er mehrzeilig ist, insgesamt *neben* die Fußnotennummer plaziert.

Die Schriftgröße, in der die Fußnote gesetzt wird, ist standardmäßig 10 Punkt; sie läßt sich aber wahlweise auf zwei Arten beeinflussen. Die erste Möglichkeit besteht darin, daß Sie mit

`\font\footfont=`*fontname*

den Zeichensatz einstellen, in dem die Fußnote gesetzt werden soll, und mit

`\fbaselineskip=`*skip*

den Zeilenabstand, der bei mehrzeiligen Fußnoten angewandt werden soll. Die Möglichkeit hat den Nachteil, daß bei Umschaltung auf eine andere Schriftart dafür wieder die Standardgröße verwendet wird.

Viel schöner ist daher die zweite Möglichkeit (die erste wurde auch nur wegen der Kompatibilität zu älteren NICE-Versionen erhalten): Definieren Sie mit

`\let\footnotestyle=\`*stilmakroname*

eine generelle Schriftgröße für die Fußnoten. Dabei ist *stilmakroname* der Name eines der im Abschnitt 2.2 angegebenen Makros für die Wahl einer anderen Schriftgröße. Der Zeilenabstand für mehrzeilige Fußnoten wird dann ebenfalls nach dem durch das Makro definierten Zeilenabstand gesetzt.

Übrigens können beim Einsatz von NICE Fußnoten auch aus mehreren Absätzen bestehen, was zwar wahrscheinlich nur in seltenen Fällen gebraucht wird, aber dann sehr nützlich sein kann.

2.4. Gestaffelte Listen

Gestaffelte Listen sind generell zweispaltige Tabellen, wobei die erste Spalte irgendwelche *Schlüsseltexte* oder *-zeichen* enthält und die zweite Spalte einen beliebigen Text, der auch länger sein und sich über mehrere Zeilen erstrecken kann. Der Schlüsseltext wird linksbündig gesetzt; die Spalte für den Schlüsseltext ist dabei so breit wie der längste Schlüsseltext in der ganzen Liste. Der Text in der zweiten Spalte wird mit Randausgleich

[12] Ja, ich weiß, das `\footnote`-Makro gibt's auch schon in plainTEX. Siehe zu diesem Problem Abschnitt 3.

gesetzt; die Spalte ist dabei so breit, daß insgesamt die ganze Zeilenlänge ausgenutzt werden kann.

Eine gestaffelte Liste wird begonnen mit

`\staggered{`*text*`}`

Hierbei ist *text* einfach der längste Eintrag plus einen Zusatzabstand zur zweiten Spalte. Der längste Eintrag wird also nicht automatisch ermittelt, sondern Sie müssen ihn explizit am Beginn der Liste angeben. Hierbei ist allerdings nur wesentlich, welche Länge der *text* hat, wenn Sie also die Länge des längsten Eintrags kennen, können Sie diese auch unmittelbar mit `\kern`*dimension* angeben. Ein Listeneintrag wird dann mit

`\key{`*schlüsseltext*`}` *fortlaufender text*

begonnen. Der *fortlaufende text* darf beliebig lang sein und auch Absätze oder andere Konstrukte enthalten. Da gestaffelte Listen schachtelbar sind, darf der fortlaufende Text wieder eine gestaffelte Liste enthalten. Das `\par`-Makro sollte jedoch nicht verwendet werden, da sonst die Listenverwaltung durcheinander kommt. Verwenden Sie einfach eine Leerzeile, um einen Absatz zu erzeugen. Der *fortlaufende text* geht bis zum nächsten Listeneintrag oder bis zum Ende der Liste.

`\staggered` setzt übrigens `\parskip=0pt plus 0.5pt`, Sie können natürlich einen anderen Abstand für Absätze unmittelbar nach dem `\staggered`-Kommando wählen. Eine gestaffelte Liste wird mit

`\endstaggered`

beendet.

Besondere gestaffelte Listen sind bereits in NICE eingebaut, vielfach wird nämlich eine gestaffelte Liste nicht in der obigen allgemeinen Form benötigt, sondern eine der im folgenden beschriebenen speziellen Versionen, die alle bestimmte Schlüsseltexte vor jedem Eintrag automatisch vorgeben. Alle Einträge in diese Listen werden mit

`\next`

begonnen; ein Schlüsseltext muß hier natürlich nicht mehr angegeben werden.

`\numbered` *n*

Jeder Schlüsseltext besteht aus einer laufenden Nummer, gefolgt von einem "Punkt". Der erste Eintrag erhält die Nummer 1. *n* ist die für die Nummer jeweils zu reservierende Stellenzahl. Bei bis zu 9 Einträgen wählt man also sinnvollerweise *n*=1, bei weniger als 100 Eintragen *n*=2, bei noch mehr *n*=3.[13] Die Liste wird mit `\endnumbered` abgeschlossen.

[13] Mehr als 999 Einträge werden wohl in einer solchen Liste nicht vorkommen.

`\itemlist{`*allkey*`}`	Der Schlüsseltext eines jeden Eintrags ist der gleiche, nämlich *allkey*. Die Liste wird mit `\enditemlist` beendet.
`\pnumbered`	Wie bei `\numbered` werden die Einträge bei 1 beginnend durchnumeriert. Allerdings wird die Nummer einfach in runde Klammern eingeschlossen. Die Liste wird – wer hätte das gedacht – mit `\endpnumbered` abgeschlossen.
`\prnumbered`	Funktioniert wie `\pnumbered`, die Numerierung erfolgt jedoch mit kleinen römischen Ziffern. Abschluß mit `\endprnumbered`.
`\alphanumbered`	Die Einträge werden mit "a)", "b)" usw. durchnumeriert. Abgeschlossen wird die Liste mit `\endalphanumbered`.

Nun ist wieder mal ein Beispiel fällig. Betrachten Sie nochmals die obige Aufstellung. Zum Vergleich ist hier die TEX-Eingabe, mit der sie erzeugt wurde:

```
\staggered{\tt \\itemlist$\lbrace$\it allkey$\rbrace$\quad}
\key{\tt \\numbered \it n} Jeder Schl\"usseltext
[...]
mit |\endnumbered| abgeschlossen.
\key{\tt \\itemlist$\lbrace$\it allkey$\rbrace$}
[...]
wird mit |\enditemlist| beendet.
\key{\tt \\pnumbered} Wie bei
[...]
mit |\endpnumbered| abgeschlossen.
[...]
\endstaggered
```

Noch ein Beispiel: Die Aufstellung am Anfang dieses Abschnittes wurde so erzeugt:

```
\itemlist {$\bullet$\quad}
\next Verschiedene {\bf Schriftgr\"o\ss{}en} [...]
\next {\bf Fu\ss{}noten}, [...]
[...]
\enditemlist
```

2.5. Verbatim-Modus

Der Verbatim-Modus ist schon im TEXbook beschrieben als ein spezieller Textsatzmodus, in dem alle Zeichen genau so gesetzt werden, wie sie in der TEX-Eingabedatei stehen. Der Verbatim-Modus ist im Prinzip so implementiert, wie es auch im TEXbook vorgeschlagen wird.

Es gibt zwei Arten des Verbatim-Modus. Mit der ersten Art können ganze Textteile so

gesetzt werden. Dieser Modus wird eingeschaltet mit

```
\beginverbatim
```

und beendet mit

```
\endverbatim
```

Diese beiden Makros sollten jeweils auf einer eigenen Zeile stehen. Alles, was zwischen diesen beiden Makros steht, wird so gesetzt, wie es in der Eingabedatei steht, und zwar im font `\verbatimfont` (standardmäßig `cmtt10`) mit Zeilenabstand `\vbaselineskip` (standardmäßig gleich 12 Punkt). Wurde allerdings mit

`\let\verbatimstyle=`*stilmakroname*

eine andere Schriftgröße für den Verbatim-Modus definiert, so wird mit der Schriftart `\tt` der eingestellten Schriftgröße gesetzt.

Beachten Sie bitte, daß bei zu langen Zeilen der Eingabedatei ein Zeilenumbruch erfolgen kann, der dann jedoch mit mit "`underfull \hbox`" gemeldet wird.

Die zweite Art des Verbatim-Modus besteht in der direkten Umschaltung auf diesen Modus mitten im laufenden Text. Die Umschaltung geschieht mit

```
|
```

Der folgende Text wird dann im Verbatim-Modus gesetzt, bis wieder ein "|"-Zeichen kommt. In einem solchen Verbatim-Textteil kann sowohl ein Zeilenumbruch als auch eine Trennung erfolgen. Diese Methode zum Gebrauch des Verbatim-Modus kann nicht in Paramtern von Makros verwendet werden.[14]

2.6. Querverweise

In einem längeren Text wird man wohl kaum ohne Verweise auf zwar im gleichen Text, aber an anderer Stelle behandelte Themen auskommen.[15] Üblicherweise verweist man entweder auf die Seitennummer oder auf die Nummer des betreffenden Abschnitts. Das von Hand zu verwalten, wäre begreiflicherweise ein Problem, denn während der Entwicklung eines Textes sind die Abschnitts-, noch mehr aber die Seitennummern einer ständigen Veränderung unterworfen.

Mit NICE geht's automatisch, und wir werden uns jetzt ansehen, mit welchen Kommandos das erreicht werden kann. Zunächst eine allgemeine Darstellung des verwendeten Verfahrens. Wenn irgendeine Textstelle referenziert werden soll, muß diese Textstelle markiert werden. Die Abschnitts- und die Seitennummer, an der die Markierung steht, werden

[14] Darum geht die temporäre Umschaltung auf den Verbatim-Modus mit "|" auch in Fußnoten nicht.

[15] Auch in dieser Anleitung wird reichlich von diesen Querverweisen Gebrauch gemacht.

in eine Datei geschrieben. Die Verwaltung dieser Datei geschieht ähnlich wie beim Inhaltsverzeichnis mit einer Zweiphasen-Methode: Die Referenz-Datei wird am Textanfang eingelesen und die während der Übersetzung definierten Referenzen werden wieder auf die Datei geschrieben. Daher muß am Ende der Textentwicklung der Text zweimal übersetzt werden, damit die Referenzen garantiert richtig sind.

Um die Verwendung von Querverweisen überhaupt zu ermöglichen, muß irgendwo am Anfang des Dokuments

`\userefs`

angegeben werden. Dadurch wird eine eventuell vorhandene Referenz-Datei[16] eingelesen und während des Übersetzungsvorgangs definierte Referenzen werden auf diese Datei geschrieben.

Die Definition eines Querverweises erfolgt mit

`\label{`*name*`}`

Abschnittsnummer[17] und Seitennummer, auf der sich die Definition befindet, werden unter dem angegebenen *name*n gespeichert. Es erfolgt keine Überprüfung, ob *name* schon definiert ist; eine weitere Definition überschreibt die alte. Sie sollten einen Querverweis nicht unmittelbar nach Verwendung von `\section`, `\subsection`, `\subsubsection` oder allgemein nach einem `\nobreak`-Befehl verwenden, da dadurch die Wirkung von `\nobreak` verlorengeht. Bei Abschnittsüberschriften bedeutet das, daß nicht mehr garantiert ist, daß nach der Überschrift nicht umgebrochen wird.

Querverweise werden referenziert mit

`\sref{`*name*`}`

das die Abschnittsnummer des referenzierten Namens in den Text einfügt, und mit

`\pref{`*name*`}`

das die Seitennummer des referenzierten Namens in den Text einfügt. In beiden Fällen gilt: Ist der betreffende Name nicht definiert – was bei der ersten Übersetzung nach erstmaliger Definition durch die Zweiphasenmethode immer der Fall ist – wird eine Meldung auf dem Bildschirm ausgegeben; in den Text wird dann der undefinierte Name in eckigen Klammern eingefügt.

2.7. Mathematiksymbole im laufenden Text

Mitunter tauchen im laufenden Text häufiger Mathematiksymbole wie “<”, “>”, “|”,

[16] mit Extension CRF

[17] Genauer: Je nachdem, ob sich die Definition in einem Abschnitt, einem Unterabschnitt oder einem Unterunterabschnitt befindet, wird die Abschnitts-, Unterabschnitts- oder Unterunterabschnittsnummer verwendet.

"{" und "}" auf. Nun ist es ziemlich umständlich, alleine zur Erzeugung dieser Zeichen immer wieder auf den Mathematikmodus umschalten zu müssen. Wenn also der Fall eintritt, daß in Ihrem Text immer wieder diese Mathematiksymbole ohne Zusammenhang mit mathematischen Ausdrücken auftauchen, sollten Sie das Makro

```
\usemathsymbols
```

benutzen.

Damit können Sie im ganzen folgenden Text (bzw. innerhalb der aktuellen Gruppe) auch außerhalb vom Mathematikmodus

- `<` zur Erzeugung von <
- `>` zur Erzeugung von >
- `|` zur Erzeugung von |
- `\{` zur Erzeugung von {
- `\}` zur Erzeugung von }

verwenden. Die temporäre Umschaltung in den Verbatim-Modus mit "|" ist dann allerdings nicht mehr möglich.

2.8. Formatieren von Programmcode

Nicht nur bei Informatikern, die ihre Arbeiten mit TEX schreiben, wird vielleicht mal das Problem auftauchen, einen Programmtext zu setzen. Zur Lösung dieses Problems gibt es zwei Möglichkeiten: Die einfachste ist, das Programm im Verbatim-Modus abzudrucken. Das geht zwar sehr schnell, ergibt aber nicht gerade ein ansprechendes Ergebnis.

Die andere Möglichkeit ist in plainTEX eine mühsame Formatierung mit Tabellensatz. Um das nun zu erleichtern, gibt es in NICE das Makro

```
\program
```

Damit wird angezeigt, daß nun ein Programmtext folgt, der folgendermaßen aufgebaut ist:

- Jede Programmzeile endet mit `\cr`.
- Eine Programmzeile darf beliebigen Text enthalten (auch Umschaltungen in den Mathematikmodus).
- Mit

  ```
  \+
  ```

  können Einrückungen definiert werden: Alle folgenden Programmzeilen bis zum Auftauchen von `\-` werden bis zu der Position eingerückt, an der das `\+` stand.
- Das Makro

  ```
  \-
  ```

macht das zuletzt verwendete `\+` wieder rückgängig. Vor dem `\-` darf nichts in der Programmzeile stehen außer eventuell weitere `\-`.

– Wird unmittelbar nach dem `\program`-Makro `\obeylines` angegeben, so kann man sich das `\cr` nach jeder Programmzeile sparen; in diesem Fall wird jede Zeile der Datei als Programmzeile aufgefaßt.

Das Ende des zu setzenden Programmtextes wird mit

```
\endprogram
```

angezeigt. Zu diesem Zeitpunkt müssen alle durch `\+` definierten Einrückungen wieder durch `\-` aufgelöst sein.

Das folgende Beispiel wird Sie davon überzeugen, daß es sich lohnt, für die Programmformatierung ein klein wenig mehr Zeit zu investieren. Das Ergebnis der folgenden TEX-Eingabe

```
\program\obeylines
{\bf begin}
\qquad\+ $Q := \{v_1\}$; markiere $v_1$;
{\bf while} $Q \neq \emptyset$ {\bf do begin}
\qquad\+ w\"ahle $v\in Q$ beliebig (BFS/DFS);
$Q := Q \setminus \{v\}$;
{\bf for all} $v' \in A(v)$ {\bf do}
\qquad\+ {\bf if} $v'$ nicht markiert {\bf then begin}
\qquad\+ $Q:=Q\cup\{v'\}$;
markiere $v'$;
\- {\bf end};
\-\- {\bf end};
\-{\bf end};
\endprogram
```

sehen Sie in Abbildung 2.

2.9. Numerierung von Abbildungen

Ebenso wie die Einrichtung der Querverweise für längere Texte sehr nützlich sein kann, ist auch eine referenzierbare Numerierung von Abbildungen für Texte mit vielen Abbildungen äußerst wertvoll. Die Referenzierung von Abbildungen läuft über die gleiche Datei, in der auch die Querverweise abgelegt werden, d. h. die Hinweise zur Zweiphasen-Methode treffen hier ebenfalls zu.

Sie müssen auch am Anfang des Dokuments

```
\userefs
```

angeben, um die automatsiche Numerierung zu verwenden. Am Ende einer durch `\midin-`

```
begin
    Q := {v1}; markiere v1;
    while Q ≠ ∅ do begin
        wähle v ∈ Q beliebig (BFS/DFS);
        Q := Q \ {v};
        for all v' ∈ A(v) do
            if v' nicht markiert then begin
                Q := Q ∪ {v'};
                markiere v';
            end;
    end;
end;
```

Fig. 2: Eine Programmformatierung mit `\program`

`sert`, `\topinsert` oder `\pageinsert` begonnenen Abbildung, also unmittelbar vor dem `\endinsert`, verpaßt man der Abbildung mit

`\figure{`*name*`}{`*titel*`}`

einen *name*n, über den sie später referenziert werden kann, und einen *titel*. Es wird dann an dieser Stelle eine zentrierte Zeile der Form

Fig. *nummer*: *titel*

in Fettschrift erzeugt. Der vergebene *name* kann übrigens bei Querverweisen schon einmal verwendet worden sein, ohne daß es zu Konflikten kommt, möglichst jedoch nicht bei vorhergehenden Abbildungen.

2.10. Einrahmen von Texten

Schöne Hervorhebungseffekte lassen sich erzielen, indem man Texte einrahmt. Das allgemeine Makro für diesen Zweck ist

`\gbox{`*rt*`}{`*vs*`}{`*hs*`}{`*text*`}`

Der angegebene *text*, der auch länger sein und Absätze enthalten kann, wird in eine `\vbox` gesetzt. Um diese Box wird ein Rahmen mit Dicke *rt* gezogen, und zwar in einem Abstand von *hs* links und rechts und *vs* oben und unten.

Um nun einen Text einzurahmen, der – einschließlich Rahmen – die ganze Zeilenbreite ausnutzen soll, benutzt man

`\boxed{`*text*`}`

Der angegebene *text* wird wieder mit einem Rahmen umgeben, der die Dicke `\rulethickness` hat (standardmäßig 0.4pt); der Rahmen wird in einer Entfernung von `\boxedkern` (standardmäßig 12pt) nach allen Richtungen gesetzt.

2.11. Sonstige Makros

Es folgt nun noch ein Überblick über weitere kleine, aber feine Makros.

`\include{`*name*`}`

fügt die Datei *name* in den Text ein,[18] falls sie existiert. Falls nicht, wird eine entsprechende Meldung ausgegeben und mit der Übersetzung fortgefahren.

`\newpage`

erzwingt einen Seitenumbruch. Beachten Sie bitte, daß mehrere `\newpage`-Befehle unmittelbar hintereinander als *ein* Befehl angesehen werden. Eine Leerseite kann jedoch etwa mit der Befehlsfolge

`\newpage \ \newpage`

erzeugt werden.

Das Makro

`\\`

erzeugt einfach einen Backslash im laufenden Text und enthebt einen so von der Notwendigkeit, immer in den Mathematik-Modus umschalten zu müssen, wenn man so ein Zeichen haben will.

Normalerweise wird Text im Blocksatz, d. h. mit Randausgleich gesetzt. In plainTEX gibt es bereits `\raggedright`, das auf linksbündigen Textsatz umschaltet. Nun können Sie auch mit

`\raggedcenter`

auf zentrierten Textsatz umschalten. Es empfiehlt sich, diesen Befehl nur innerhalb einer Gruppe zu geben, da sonst der zentrierte Textsatz bis zum Textende gilt, wenn er nicht anderweitig wieder rückgängig gemacht wird.

Um Text etwas "schmäler" zu setzen, d. h. links und rechts um einen gewissen Wert eingerückt, gibt es

`\narrower`

das wie das gleichnamige Makro in plainTEX funktioniert, nur daß nicht um den Wert von `\parskip` eingerückt wird, sondern um den Wert eines speziellen Registers `\narrowskip`. Standardwert ist hier 20 pt. Einen anderen Wert für die Einrückung setzen Sie also mit

`\narrowskip=`*dimension*

[18] und entspricht insoweit dem `\insert`-Makro.

Soll sich die Einrückung nur auf den linken Rand beziehen, verwenden Sie

`\lnarrower`

statt `\narrower`. In beiden Fällen gilt wieder die Empfehlung, die Makros nur innerhalb einer Gruppe zu verwenden, da sonst ihre Wirkung nicht mehr auf einfache Weise (durch Verlassen der Gruppe) rückgängig gemacht werden kann.

Für manche Anwndungen will man zwar einen Absatz, aber nicht den Abstand `\parskip`, den dieser Absatz mit sich bringt. Umständlich ist, `\parskip` jedesmal in einer solchen Situation auf Null zu setzen und dann wieder auf den alten Wert zu setzen. Darum gibt es

`\parwithoutskip`

das einen Absatz mit Abstand `0pt plus 0.5pt` erzeugt. Die Streckkomponente ist erforderlich, damit sich TEX beim Seitenumbruch etwas leichter tut. Mit `\parwithoutskip` wird auch gleich ein neuer Absatz begonnen, es sollte also nach dem Makro auch wirklich Text (im Horizontalmodus) folgen.

Nützlich für die eindeutige Kennzeichnung von Versionen eines Textes ist das Makro

`\date`

das das aktuelle Tagesdatum in der Form *tag/monat/jahr* erzeugt, wobei *tag* und *monat* jeweils zweistellig, *jahr* vierstellig gesetzt werden.

Last not least

`\ifndef{`*macroname*`}`

das testet, ob *macroname* bereits definiert ist. `\ifndef` kann wie eine normale `\if`-Abfrage verwendet werden, d. h. nach `\ifndef` kommt das, was abgearbeitet werden soll, wenn *macroname* bereits definiert ist, dann schreibt man ggf. `\else`, gefolgt von den im negativen Falle auszuführenden Kommandos. Abgeschlossen wird eine `\ifndef`-Sequenz wie gewohnt mit `\fi`.

3. Kompatibilität mit plainTEX

Wir kommen nun zu einer heiklen Frage, nämlich: Wie kompatibel ist NICE zu plainTEX? Die einfache Antwort: Fast hundertprozentig. Dieses “fast” zu erläutern ist Zweck dieses Abschnitts.

Die offensichtlichste Inkompatibilität ist durch das neue `\footnote`-Makro bedingt. Dieses Makro gibt es auch schon in plainTEX, allerdings mit zwei Parametern. In NICE wird die Fußnotennummer automatisch erzeugt, darum besteht nur noch die Notwendigkeit für einen Parameter, den Fußnotentext selbst. Glücklicherweise gibt es für dieses

Dilemma eine Lösung, unter NICE ist nämlich das alte `\footnote`-Makro von plainTEX über `\plainfootnote` zugänglich. Wer also das alte `\footnote`-Makro auch unter NICE weiterverwenden möchte, setzt am Textanfang

```
\let\nfootnote=\footnote \let\footnote=\plainfootnote
```

ein. Dies hat den Effekt, daß man sich immerhin noch mit `\nfootnote` numerierte Fußnoten erzeugen kann.

Daß auch am `\vfootnote`-Makro leichte Änderungen vorgenommen wurden, ist bereits im Abschnitt 2.3 angedeutet worden: Der Text der Fußnote wird um `\footnoteindent` eingerückt, außerdem wird der mit `\footfont` bzw. `\footnotestyle` eingestellte Zeichensatz verwendet. Wer hier unbedingt das ursprüngliche Verhalten haben möchte, sagt

```
\let\vfootnote=\ovfootnote
```

am Textanfang.

Das `\narrow`-Makro wurde – wie im Abschnitt 2.11 bereits bemerkt – ebenfalls vom aktuellen Wert für `\parindent` unabhängig gemacht. Auch hier gilt wieder:

```
\let\narrow=\onarrow
```

stellt die Originalwirkung wieder her.

Die Zeichenklasse von "|" wurde auf `\active` gesetzt, um damit die temporäre Umschaltung in der Verbatim-Modus zu ermöglichen. Um "|" normal im laufenden Text verwenden zu können, müssen Sie die Zeichenklasse wieder auf "other" setzen:

```
\catcode`|=12
```

Die sonstigen Änderungen gegenüber plainTEX beziehen sich auf veränderte Werte für Parameter, über die die Tabelle in der Abbildung 3 Auskunft gibt.

4. Konfiguration von NICE

Dieser Abschnitt beschreibt, wie Sie NICE auf Ihrem Rechner installieren, und wie Sie weitergehende Änderungen am von NICE normalerweise erzeugten Textlayout vornehmen können, die über die üblicherweise benutzten Möglichkeiten hinausgehen.

4.1. Erzeugung einer Formatdatei

Sie können NICE einfach benutzen, indem Sie das Makropaket mit

```
\input nice
```

Parameter	*Wert in*	
	plain TEX	*NICE*
\vsize	8.9 true in	9.2 true in
\hsize	6.5 true in	6.5 true in
\lineskip	1pt	0pt
\smallskipamount	3pt plus 1pt minus 1pt	6pt plus 1pt minus 1pt
\medskipamount	6pt plus 2pt minus 2pt	12pt plus 2pt minus 2pt
\bigskipamount	12pt plus 4pt minus 4pt	18pt plus 3pt minus 3pt
\parskip	0pt plus 1pt	6pt plus 1pt minus 1pt
\parindent	20pt	0pt
\tolerance	200	10000
\hbadness	1000	5000

Fig. 3: Geänderte TEX-Parameter

am Textanfang in Ihren Text einlesen. Das dauert aber jedesmal eine Weile.[19] Die Alternative dazu wird im folgenden beschrieben.

Bekanntlich lädt TEX zu Beginn des Übersetzungsvorgangs die Formatdatei `plain.fmt`, falls Sie nicht mit "&" auf der Aufrufzeile eine andere Formatdatei vorgeben. Wenn Sie mit NICE häufiger arbeiten, sollten Sie sich eine eigene Formatdatei, die das NICE-Paket schon enthält, erzeugen. Dies geht wie folgt:

a) Kopieren Sie die Datei `nice.tex` (das Makropaket) in ein Verzeichnis, wo es beim Aufruf von TEX gefunden werden kann, etwa in das Verzeichnis, wo schon Ihre Makrodateien stehen.

b) Wechseln Sie in das Verzeichnis, in dem die Formatdateien stehen, und erstellen Sie dort die folgende Datei mit Namen `nicee.tex` (wenn Sie mit englischer Trennung arbeiten) bzw. `niceg.tex` (wenn Sie mit deutscher Trennung arbeiten):

```
\input plain
\input nice
% input any local modifications here
\dump
```

c) Geben Sie auf Ihrem Rechner das Kommando

```
initex nicee
```

bzw.

```
initex niceg
```

[19] Wie lange die "Weile" ist, hängt natürlich stark von Ihrem Rechner ab.

und die Datei `nicee.fmt` bzw. `niceg.fmt` wird erzeugt.

4.2. Lokale Änderungen

Änderungen jeglicher Art, die mit NICE zusammenhängen, und die Sie dauerhaft für Ihre Installation einstellen wollen, nehmen Sie bitte *keinesfalls* im NICE-Paket selbst (also in `nice.tex`) vor. Tragen Sie Ihre Änderungen in obige Datei `nicee.tex` bzw. `niceg.tex` vor der `\dump`-Zeile ein. Sie sollten jedoch keine Änderungen vornehmen, die mit Ihrer lokalen Installation erstellte Texte inkompatibel zu der unveränderten NICE-Version machen.

Bei Rechnersystemen, in denen mehrere Benutzer auf NICE zugreifen, sollten überhaupt keine lokalen Änderungen auf diese Weise vorgenommen werden. Am besten ist es hier, wenn sich jeder Benutzer seine Änderungen in eine Makrodatei schreibt und diese zusätzlich zu NICE einliest.[20]

4.3. Das Abschnitt-Numerierungsschema

Das standardmäßige Numerierungsschema für Abschnitte und ihre Untergliederungen kennen Sie bereits. Es lautet

abschnittsnummer.␣

für Abschnitte,

abschnittsnummer.unterabschnittsnummer.␣

für Unterabschnitte, und

abschnittsnummer.unterabschnittsnummer.unterunterabschnittsnummer.␣

für Unterunterabschnitte. Dabei bezeichnet "␣" eine Leerstelle. Beachten Sie bitte, daß die *abschnittsnummer* im Anhang ein Großbuchstabe ist.

Das Format dieses Numerierungsschemas können Sie nun verändern. Die Schemata für Abschnitte, Unterabschnitte und Unterunterabschnitte befinden sich in den Tokenregistern

`\sectionnnscheme`

bzw.

`\subsectionnnscheme`

bzw.

`\subsubsectionnnscheme`

Die Standardbelegung für diese Register ist

`\gensect.␣`

[20] Es kann sich natürlich auf jeder Benutzer selbst eine eigene Formatdatei mit seinen Änderungen erzeugen.

für `\sectionnscheme`,

```
\gensect.\number\subsectionno._
```

für `\subsectionnscheme`, und

```
\gensect.\number\subsectionno.\number\subsubsectionno._
```

für `\subsubsectionnscheme`. Dabei erzeugt `\gensect` jeweils die Abschnittsnummer; dafür ist ein eigenes Makro erforderlich, weil normalerweise eine Nummer erzeugt werden muß und ein Großbuchstabe, wenn man sich im Anhang befindet.

Durch Ändern dieser Tokenregister ändern Sie das Numerierungsschema, und zwar nicht nur für die Überschriften von Abschnitten bzw. ihrer Untergliederungen, sondern auch im Inhaltsverzeichnis und in der Kopfzeile. Jedoch wird die Numerierung nicht bei Querverweisen geändert; hier müssen Sie das `\label`-Makro direkt umdefinieren.

A. Reservierte Steuersequenzen

Dieser Abschnitt ist eine vollständige Übersicht über die internen und extern zugänglichen Makros und sonstige Steuersequenzen, die in NICE definiert werden, und die auf der obersten Ebene sichtbar sind (bei Verwendung einiger Befehle kommt man in eine neue Gruppe, in der dann weitere Makros definiert werden, die beim Verlassen der Gruppe aber wieder gelöscht werden; diese sind hier nicht aufgeführt). In Klammern ist ggf. die Nummer des Abschnitts aufgeführt, in dem die entsprechende Steuersequenz näher erläutert wird. In der Spalte "X" steht ein "i", falls das Makro *nur* NICE-intern ist und keinesfalls in Ihren Texten verwendet werden sollte, da nicht garantiert ist, daß das Makro in der nächsten Version noch existiert oder noch genauso arbeitet. Steht hingegen dort ein "e", so ändert sich das Makro auch in den Folgeversionen von NICE nicht.

Steuersequenz	*Typ*	*X*	*Bedeutung*
\\	Makro	e	erzeugt Backslash (2.11)
`\alphanumbered`	Makro	e	erzeugt alphanumerische Liste (2.4)
`\appendix`	Makro	e	erzeugt Anhang (1.2)
`\article`	Makro	e	setzt Kopfzeile (1.4)
`\beginverbatim`	Makro	e	schaltet Verbatim-Modus ein (2.5)
`\book`	Makro	e	setzt Kopfzeile (1.4)
`\boxed`	Makro	e	erzeugt eingerahmten Text (2.10)
`\boxedkern`	Dimension	e	Rahmenabstand vom Text (2.10)
`\contentslevel`	Zähler	e	Information im Inhaltsverzeichnis (1.3)
`\contfill`	Tokens	e	Füllmuster für Inhaltsverzeichnis (1.3)
`\contskip`	Skip	e	Abschnittsabstand im Inhaltsverzeichnis (1.3)
`\d@specials`	Makro	i	benötigt für Verbatim-Modus
`\date`	Makro	e	setzt Datum ein (2.11)
`\documenttitlefont`	Font	e	Zeichensatz für Titelzeile (1.1)
`\doverbatim`	Makro	i	benötigt für Verbatim-Modus
`\eightbf`	Font	e	cmbx8

\eighti	Font	e	cmmi8
\eightit	Font	e	cmti8
\eightpt	Makro	e	stellt Schriftgröße 8pt ein (2.2)
\eightrm	Font	e	cmr8
\eightsl	Font	e	cmsl8
\eightsy	Font	e	cmsy8
\eightt	Font	e	cmtt8
\escapeit	Makro	i	benötigt für Querverweise
\fbaselineskip	Skip	e	Zeilenabstand von Fußnoten (2.3)
\fcontents	Ausgabedatei	e	benötigt für Inhaltsverzeichnis
\figure	Makro	e	Betitelung einer Abbildung (2.9)
\fivept	Makro	e	stellt Schriftgröße 5pt ein (2.2)
\footfont	Font	e	Zeichensatz für Fußnoten (2.3)
\footno	Zähler	e	enthält Nummer der letzten Fußnote
\footnote	Makro	e	Setzen einer Fußnote (2.3)
\footnoteindent	Dimension	e	Einrückung für Fußnote (2.3)
\fref	Makro	e	Referenzierung von Abbildungen (2.9)
\frefs	Ausgabedatei	e	benötigt für Querverweise
\gbox	Makro	e	erzeugt eingerahmten Text (2.10)
\genheading	Makro	e	erzeugt Überschriftenzeile
\gensect	Makro	e	erzeugt Abschnittsnummer
\hdrule	Makro	i	erzeugt Linie für Kopfzeile
\hdrulepos	Dimension	e	Höhenoffset der Linie für Kopfzeile
\ifinb@x	If	i	benötigt zum Setzen von Programmcode
\ifndef	Makro	e	prüft Makrodefinition (2.11)
\inb@xfalse	If	i	benötigt zum Setzen von Programmcode
\inb@xtrue	If	i	benötigt zum Setzen von Programmcode
\include	Makro	e	Einfügen einer Datei in den Text (2.11)
\itemlist	Makro	e	erzeugt eine gestaffelte Liste (2.4)
\itemno	Zähler	e	zählt Einträge in numerierte Listen (2.4)
\label	Makro	e	markiert Ziel eines Querverweises (2.6)
\lnarrower	Makro	e	links verengter Textsatz (2.11)
\makecontents	Makro	e	Erzeugung eines Inhaltsverzeichnisses (1.3)
\makevoidcontents	Makro	e	benötigt für Inhaltsverzeichnis
\marklevel	Zähler	e	legt Information in Kopfzeile fest (1.4)
\narrower	Makro	e	links und rechts verengter Textsatz (2.11)
\narrowskip	Skip	e	Größe des verengten Textsatzes (2.11)
\newpage	Makro	e	erzeugt neue Seite (2.11)
\nice@mgt	Zeichen	i	Synonym für >-Zeichen
\nice@mlbr	Zeichen	i	Synonym für {-Zeichen
\nice@mlt	Zeichen	i	Synonym für <-Zeichen
\nice@mrbr	Zeichen	i	Synonym für }-Zeichen
\nice@mvert	Zeichen	i	Synonym für |-Zeichen
\niceversion	Makro	e	setzt NICE-Versionsnummer ein
\ninebf	Font	e	cmbx9
\ninei	Font	e	cmmi9
\nineit	Font	e	cmti9
\ninept	Makro	e	stellt Schriftgröße 9pt ein (2.2)
\ninerm	Font	e	cmr9
\ninesl	Font	e	cmsl9
\ninesy	Font	e	cmsy9
\ninett	Font	e	cmtt9
\numbered	Makro	e	erzeugt numerierte Liste (2.4)
\onarrower	Makro	e	\narrower-Makro von plainTEX (2.11)

\ovfootnote	Makro	e	\vfootnote-Makro von plainTeX (2.3)
\parwithoutskip	Makro	e	erzeugt Absatz ohne Abstand (2.11)
\pindent	Dimension	i	benötigt für den Satz von Programmcode
\plainfootnote	Makro	e	\footnote-Makro von plainTeX (2.3)
\plainsection	Makro	e	erzeugt Abschnitt ohne Numerierung (1.2)
\pnumbered	Makro	e	erzeugt numerierte Liste (2.4)
\ppindent	Dimension	i	benötigt für den Satz von Programmcode
\pr@gr@m	Makro	i	benötigt für den Satz von Programmcode
\pr@gram	Makro	i	benötigt für den Satz von Programmcode
\pref	Makro	e	Querverweis auf Seitennummer (2.6)
\prnumbered	Makro	e	erzeugt eine römisch numerierte Liste (2.4)
\program	Makro	e	Formatierter Satz von Programmcode (2.8)
\raggedcenter	Makro	e	Schaltet auf zentrierten Textsatz (2.11)
\rulethickness	Dimension	e	Rahmendicke bei eingerahmten Texten (2.10)
\scindent	Dimension	e	Einrückung f. Abschnitte im Inhaltsverz. (1.3)
\scstyle	Makro	e	Stil f. Abschnitte im Inhaltsverz. (1.3)
\section	Makro	e	erzeugt numerierten Abschnitt (1.2)
\sectionno	Zähler	e	Nummer des aktuellen Abschnitts
\sectionnscheme	Tokens	e	Numerierungsformat für Abschnitte (4.3)
\sectiontitlefont	Font	e	Zeichensatz für Abschnittsüberschriften
\setupverbatim	Makro	i	benötigt für Verbatim-Modus
\sevenit	Font	e	cmti7
\sevenpt	Makro	e	stellt Schriftgröße 7pt ein (2.2)
\sixbf	Font	e	cmbx6
\sixi	Font	e	cmmi6
\sixpt	Makro	e	stellt Schriftgröße 6pt ein (2.2)
\sixrm	Font	e	cmr6
\sixsy	Font	e	cmsy6
\spvert	Makro	i	benötigt für Verbatim-Modus
\sref	Makro	e	Querverweis auf Abschnitt (2.6)
\sscindent	Dimension	e	Einrückung f. Unterabschnitte im Inhaltsverz. (1.3)
\sscstyle	Makro	e	Stil f. Unterabschnitte im Inhaltsverz. (1.3)
\ssscindent	Dimension	e	Einr. f. Unterunterabschn. im Inhaltsverz. (1.3)
\ssscstyle	Makro	e	Stil f. Unterunterabschn. im Inhaltsverz. (1.3)
\staggbox	Box	i	benötigt für gestaffelte Listen
\staggered	Makro	e	setzt allgemeine gestaffelte Liste (2.4)
\subsection	Makro	e	erzeugt numerierten Unterabschnitt (1.2)
\subsectionno	Zähler	e	Nummer des letzten Unterabschnitts
\subsectionnscheme	Tokens	e	Numerierungsformat für Unterabschnitte (4.3)
\subsectiontitlefont	Font	e	Zeichensatz für Unterabschnittsüberschriften
\subsubsection	Makro	e	erzeugt numerierten Unterunterabschnitt (1.2)
\subsubsectionno	Zähler	e	Nummer des letzten Unterunterabschnitts
\subsubsectionnscheme	Tokens	e	Numerierungsformat für Unterunterabschn. (4.3)
\subsubsectiontitlefont	Font	e	Zeichensatz f. Unterunterabschnittsüberschr.
\testinclude	Eingabedatei	i	benötigt für \include-Makro
\title	Makro	e	Titelsatz (2.1)
\titlepage	Makro	e	Erzeugung einer Titelseite (1.1)
\tw@num	Makro	i	benötigt für \date-Makro
\txver	Makro	i	benötigt für Verbatim-Modus
\uncatcodespecials	Makro	i	benötigt für Verbatim-Modus
\usemathsymbols	Makro	e	Satz von Mathematiksymbolen (2.7)
\userefs	Makro	e	schaltet Querverweise ein (2.6)
\vbaselineskip	Skip	e	Zeilenabstand im Verbatim-Modus (2.5)
\verbatimfont	Font	e	Zeichensatz für Verbatim-Modus (2.5)

\versionof	Makro	e	setzt Datum der NICE-Version ein
\vfootnote	Makro	e	geänderte Version von \vfootnote in plainTEX