

The *ln*T_EX Package*

Martin Thorsen Ranang
mtr@ranang.org

2013/03/12

1 Introduction

This package adds functionality to L^AT_EX that eases typesetting and indexing of phrases, acronyms and names in a consistent manner.

`intex` The really short usage description is that in order to use the package, insert
`\co` `\usepackage{intex}` at the beginning of your L^AT_EX source file. After that, you
can wrap the macro `\co{<concept>}` around any concept you want to typeset
and/or typeset in some special way.

2 Background

I have been using L^AT_EX since the spring of 1997. Since then I have written several technical documents. Like many others, I try to present my work in an accessible way to the reader, and I believe that L^AT_EX can help with the technicalities of presenting technical writing in a clear and precise way. For example, I have always tried to explain every non-trivial acronym used in my documents; include a meaningful index; and to also leave some clues to the reader through the typesetting, so that it will be easier to find the key phrases in the document.

`acronym` Already, packages exist that provide functionality that eases the acronym¹ and
`index` indexing² operations mentioned above. However, problems quickly arise when
writing about an acronym in both singular and plural. For example, let's say you
want to use the concept *informed search* (abbreviated IS). Then, if you want to
write about that concept in plural, the logical acronym would be ISes (informed
searches). At the same time, you probably want those two occurrences—perhaps
typeset several chapters apart—to be indexed as being the same concept.

*This document corresponds to `intex`, revision , dated March 12, 2013.

¹The `acronym` package, written by Tobias Oetiker, available from [CTAN:/macros/latex/contrib/acronym](#).

²The `index` package, written by David M. Jones, available from [CTAN:/macros/latex/contrib/index](#).

The *InT_EX* package was written to reduce the work needed to handle such a task. This has been done by combining the functionality of the `acronym` and the `index` packages with an external *Python*³ script.

3 Usage

How you can use *InT_EX* should be clearer after examining some examples. The central idea in *InT_EX* is that a phrase or a word worth indexing constitutes some kind of a *concept*—in a broad sense of the word. A concept can be of several kinds. It can be either an *acronym* (or abbreviation), the name of an entity (a *person*, or an organization), or of the “plain” kind (simply a phrase). Hence, we will refer to the three kinds of *concepts* as acronym, person, and plain.

In the above paragraph, the word “concepts” was defined as a concept of the *plain* kind, and it was defined to be indexed as the word “concept”. Furthermore, the words “acronym”, “person”, and “plain” were also defined as plain. However, these concepts are defined as *sub-concepts* of “concept” and should be indexed accordingly.

3.1 Package Options

`noindex` : Whether *InT_EX* should generate an index or not. (*Default: true*).

`nowarnundef` : Whether *InT_EX* should generate in-document warnings where unknown/undeclared concepts are encountered. (*Default: true*).

`nomargin⟨type⟩` : Tell *InT_EX* not to add margin notes whenever *new* concepts of kind *⟨type⟩* are typeset, where *⟨type⟩* is one of plain, acronym, or person.

3.2 Examples

<p>It is easy to refer to (and thus index) acronyms, like dihydrogen monoxide (H_2O). And sub-concepts, like dihydrogen monoxide reserve (H_2O reserve)</p>	<pre> 1 \makeatletter 2 \@itx@margin@acronymfalse% 3 \makeatother 4 It is easy to refer to (and thus index) 5 acronyms, like \co{H2O}. And 6 sub-concepts, like \co{H2O reserve} </pre>
---	---

³Python is available from <http://www.python.org/>.

We could talk about multiple identities (*IDs*), or a single identity (*ID*). The following table shows explicitly defined formatting:

Format	Expansion
forced long	“identity”
short	“ID”
full	“identity (<i>ID</i>)”

```

1 \makeatletter
2 \@itx@margin@acronymfalse%
3 \makeatother
4 We could talk about multiple \co{IDs},
5 or a single \co{ID}. The following
6 table shows explicitly defined
7 formatting:\\
8
9 \begin{tabular}{@{}ll@{}}
10 \toprule
11 Format & Expansion \\
12 \midrule
13 forced long & “\coL{ID}”\\
14 short & “\coS{ID}”\\
15 full & “\coF{ID}” \\
16 \bottomrule
17 \end{tabular}

```

The *InTeX* package also supports indexing sub-terms of acronyms, like the *Swedish Semantic Information for Multifunctional Plurilingual Lexica (Swedish SIMPLE)*, which is part of the *Semantic Information for Multifunctional Plurilingual Lexica (SIMPLE)* project.

```

1 \makeatletter
2 \@itx@margin@acronymfalse%
3 \makeatother
4 The \InTeX\ package also supports
5 indexing sub-terms of acronyms,
6 like the \co{Swedish SIMPLE}, which
7 is part of the \co{SIMPLE} project.

```

The package supports typesetting words differently in the text and in the index. For example, take a look at the definition of the term “*compound-word analyzer*” in the *InTeX*-file. Example that provokes hyphenation: a new, experimental compound-word analyzer, used for *Norwegian compound-word analysis* of *non-deverbal verb compounds*, perhaps found in *Bokmålsordboka*. Generally, analyzing *compounds*.

The word *carnivore* means $m^{e_a}_t$ eater. There are both *feline* and *canine* $m^{e_a}_t$ eaters in the animal kingdom.

```

1 The word \co{carnivore} means
2 \co{meat eater}. There are both
3 \co{feline} and \co{canine}
4 \co{meat eaters} in the animal
5 kingdom.

```

compound-word analyzer

Norwegian compound-word analysis
non-deverbal verb compounds
Bokmålsordboka
compounds

water

It is also possible to refer to acronyms from non-acronym entries, like *water*.
It is also possible to refer to a concept using capitalized words, as in

Carnivore eat meat.

```

1 \co{Carnivore} eat meat.

```

The package also supports “special” typesetting of acronyms, like

`mkintex` automatically inflects words following common patterns. For example, *index* and *indices* both refer to the same concept, even though only *index* is entered in the *LaTeX*-file. The same goes for *vertex* and *vertices*, and *dog* and *dogs*.

```
1 \verb|mkintex| automatically inflects
2 words following common patterns. For
3 example, \co{index} and \co{indices}
4 both refer to the same concept, even
5 though only \verb|index| is entered
6 in the \LaTeX-file. The same goes for
7 \co{vertex} and \co{vertices}, and
8 \co{dog} and \co{dogs}.
```

the King The *LaTeX* package handles the notion of multiple names for the same concept too. For example, a lot of people believes that *the King* is referring to one person only, and that is the late *Elvis Aaron Presley*.

3.2.1 Index Definitions

The concept index definition file used for the above paragraph looks like:

```
1 % -*- latex -*-
```

The file can be divided into different sections, according to the kind of concepts to be declared. To set the current section, use a single line that must contain exactly “% **<type>**”, where *<type>* is either **ACRONYMS**, **CONCEPTS**, or **PERSONS**.

```
2 %
3 % $Id$
4 %
5 % name=main
6 % default_inflection=singular
7 %
8 % Copyright (C) 2005--2007 by Martin Thorsen Ranang
9 %
10
```

the above line should mean that `\co{synsets}` in the text should be indexed as if it read `\co{synset}`. However, if the plural of the concept occurs first (in the [part of] document), its full-form should be *typeset* “synonym set” + “s” (indicated by the #y) in the index file. In other words, it only a short-hand notation. On the other hand, in the next definition, another short-hand notation #y is used that will transform “y” into “ies” as the end of the last word:

```
11 % In general, to refer to the full-form of a concept, use the \cof
12 % macro.
13 %
14 % *ACRONYMS*
15 AI →Artificial Intelligence
```

```

16     AIC    ↗- Complete
17
18     H2O@H$_{2}$O    ↗dihydrogen monoxide
19     - reserve
20
21     ID    ↗identity
22
23     SIMPLE    ↗\textit{Semantic Information for Multifunctional Plurilingual Lexica}    ↗:sort_as=Semantic
24     Swedish -    ↗\textit{Swedish -}
25
26     synset    ↗synonym set
27
28     % *PEOPLE*
29     EAP    ↗Presley, Elvis Aaron
30     EAPK    ↗King, the --> EAP
31     JFK    ↗Kennedy, John Fitzgerald
32     MLK    ↗King, Martin Luther
33     Madonna    ↗Madonna
34     VWvG    ↗van Gogh, Vincent Willem
35
36     % *CONCEPTS*
37     Bokmålsordboka@\textit{Bokmålsordboka}    ↗:sort_as=Bokmålsordboka
38
39     carnivore
40         feline
41             domestic cat
42             tiger
43         canine
44             Gray Wolf
45             domestic dog
46     compound@com\\-pound
47     -\word analyzer
48     non\deverbal verb -
49     Norwegian -\word analysis
50     concept
51         acronym (-)
52         person (-)
53         plain (-)
54         sub\--
55
56     dog    ↗:sort_as=sog me bold :comment=\textit{(explicitly sorted)}
57
58     idea
59     index
60
61     (meat eater)@{m$~{\textrm{e}}$a$_{\textrm{t}}$ eater} --> carnivore
62
63     synonymy
64     synonym --> synonymy
65     direct -

```

```

66     indirect -
67
68     (InTeX logo)@{\InTeX\ logo}}
69
70     Python
71
72     TeXnician@{\TeX}nician
73     -'s tool
74
75     vertex
76
77     water --> H2O
78
79     % Local Variables:
80     % mode: latex
81     % TeX-master: t
82     % ispell-local-dictionary: "american"
83     % mode: flyspell
84     % End:

```

3.3 Compilation

mkintex As mentioned earlier, the package includes an external program named **mkintex**. The typical usage of **mkintex**, given that your document is named $\langle name \rangle$, would be *concept*:

1. `latex $\langle name \rangle$.tex`
2. `mkintex $\langle name \rangle$ $\langle name \rangle$.itx -o $\langle name \rangle$.rix [-a acronyms.tex -p persons.tex]`
3. `makeindex $\langle name \rangle$`
4. `makeindex -o $\langle name \rangle$.rid $\langle name \rangle$.rix`
5. `latex $\langle name \rangle$.tex`

4 Macros

\backslash InTeX
InTeX logo This is simply a macro for typesetting the *InTeX* logo.

5 Implementation

After the customary identification,

```

1 \def\filename{intex}%
2 \ProvidesPackage{intex}[2008/10/13 v1.1
3 Support for concept, acronym, and proper-name typesetting and indexing]%

```

we continue by defining the package options.

5.1 Package Options

`noindex` Let the conditional `\if@itx@index` control whether *IntE*X should generate an index or not. The default is to perform indexing. The option `noindex` turns this feature off.

```
4 \newif\if@itx@index%
5 \@itx@indextrue%
6 \DeclareOption{noindex}{\@itx@indexfalse}%
```

`nowarnundef` The conditional `\if@itx@nowarnundef` controls whether *IntE*X should include in-document warnings about undefined concepts or not. The default is to warn about undefined concepts inside the document. The `nowarnundef` option turns this feature off.

```
7 \newif\if@itx@warn@undef%
8 \@itx@warn@undeftrue%
9 \DeclareOption{nowarnundef}{\@itx@warn@undeffalse}%
```

`nomarginplain` The conditionals `\if@itx@margin@<kind>`—where *<kind>* is one of `plain`, `acronym`, and `person`—control whether the short version of each first-occurrence of a concept (of kind *<kind>*, per significant document part) should also be typeset as a margin-label.

```
\if@itx@margin@plain
\if@itx@margin@acronym
\if@itx@margin@person
10 \newif\if@itx@margin@plain%
11 \newif\if@itx@margin@acronym%
12 \newif\if@itx@margin@person%
13 \@itx@margin@plaintrue%
14 \@itx@margin@acronymtrue%
15 \@itx@margin@persontrue%
16 \DeclareOption{nomarginplain}{\@itx@margin@plainfalse}%
17 \DeclareOption{nomarginacronym}{\@itx@margin@acronymfalse}%
18 \DeclareOption{nomarginperson}{\@itx@margin@personfalse}%
```

Next, process the options.

```
19 \ProcessOptions%
```

5.2 External Packages

`index` Now, if `\if@itx@index` is *true*, then require the package `index` to be loaded. If not, we define a handy macro usually defined in that package.

```
20 \if@itx@index%
21 \RequirePackage{index}%
22 \makeindex%
23 \newindex{raw}{rix}{rid}{Index}%
24 \else%
25 \def\@nearverbatim{\expandafter\strip@prefix\meaning}%
26 \fi%
```

`marginnote` Only require the `marginnote` package if it is really required. This is done in an attempt to avoid wasting counters.

```

27 \if@itx@margin@plain%
28   \RequirePackage{marginnote}[2006/10/26]%
29 \fi%
30 \if@itx@margin@acronym%
31   \RequirePackage{marginnote}[2006/10/26]%
32 \fi%
33 \if@itx@margin@person%
34   \RequirePackage{marginnote}[2006/10/26]%
35 \fi%

```

Please note that the `marginnote` package will only be loaded once, even if it gets required multiple times.

```

acronym  Anyhow, require the acronym and the ifthen packages to be loaded.
ifthen   36 \RequirePackage{acronym}[2008/05/28]%
          37 \RequirePackage{ifthen}%

```

5.3 The *in*TeX Logo

```

\InTeX  Define a TEX-ish logo for this package.
          38 \newcommand*\InTeX{\textsl{In}\kern-.07em\TeX}%

```

5.4 Font Definitions

The following commands define the font-selection commands used to typeset the different kinds of concepts in different situations.

```

\itxplaindeffont  These commands are used to typeset plain concepts.
\itxplainfollowfont  39 \newcommand\itxplaindeffont[1]{\emph{#1}}%
\itxplainmarginfont  40 \newcommand\itxplainfollowfont[1]{#1}%
                     41 \newcounter{itxpl}%
                     42 \newcommand\@itxbasemarginfont[1]{%
                     43   \stepcounter{itxpl}%
                     44   \ifthenelse{\isodd{\pageref{itxpl-\theitxpl}}}{%
                     45     \raggedright\hspace{0pt}\footnotesize\textsf{#1}% odd
                     46   }{%
                     47     \raggedleft\hspace{0pt}\footnotesize\textsf{#1}% even
                     48   }%
                     49   \label{itxpl-\theitxpl}%
                     50 }%
                     51 \newcommand\itxplainmarginfont[1]{%
                     52   \@itxbasemarginfont{#1}%
                     53 }%

\itxacronymdeffont  For acronyms:
\itxacronymdefshortfont  54 \newcommand\itxacronymdeffont[1]{#1}%
\itxacronymshortfont    55 \newcommand\itxacronymdefshortfont[1]{\emph{#1}}%
\itxacronymmarginfont    56 \newcommand\itxacronymshortfont[1]{#1}%
                     57 \newcommand\itxacronymmarginfont[1]{%

```



```

58 \@itxbasemarginfont{#1}%
59 %\raggedleft\hspace{Opt}\footnotesize\textsf{#1}%
60 }%

```

```

\itxpersondeffont For persons:
\itxpersonfirstfont 61 \newcommand{\itxpersondeffont}[1]{\emph{#1}}%
\itxpersonlastfont 62 \newcommand{\itxpersonfirstfont}[1]{#1}%
\itxpersonmarginfont 63 \newcommand{\itxpersonlastfont}[1]{#1}%
64 \newcommand{\itxpersonmarginfont}[1]{%
65 \@itxbasemarginfont{#1}%
66 %\raggedleft\hspace{Opt}\footnotesize\textsf{#1}%
67 }%

```

5.5 The (Low-Level) Clockwork of the Package

`co@serial` First, define a counter that is used to enumerate new concept definitions.

```
68 \newcounter{co@serial}%
```

`co@equiv@serial` First, define a counter that is used to enumerate new concept definitions.

```
69 %\newcounter{co@equiv@serial}%
```

`co@type` First, define a counter that is used to enumerate new concept definitions.

```
70 \newcounter{co@type}%
```

The `co@type` counter is used only inside the `\@itx` command.

`\itxundefcomment` Then, define the comment to display where use of undefined concepts are detected.

```
71 \newcommand*\itxundefcomment[1]{\emph{(undefined concept ‘‘#1’’)}}%
```

Define a couple of convenience macros.

```

72 \long\def\@firstofthree#1#2#3{#1}%
73 \long\def\@secondofthree#1#2#3{#2}%
74 %\newcommand*\@secondofthree[3]{#2}%

```

Make it possible to reset the “defined” flag for each concept. After a reset, the next time that concept occurs, it is typeset as if it’s the first occurrence of that concept.

```

75 \def\ITX@reset#1{%
76 \global\expandafter\let\csname itx@#1\endcsname\relax}%

```

5.5.1 Typesetting of Margin Labels

`\@itxmarginlabel` Define a macro to typeset the concepts at first-occurrence points in the margin.

```

77 \newcommand*\@itxmarginlabel[2]{%
78 \hspace{Opt}%

```

The second argument is the *identity* of the entity we’re typesetting, while the first argument signals its *type*; that is, whether we’re typesetting a...

```
79 \ifcase#1%
```

```

...plain concept, ...
80   \if@itx@margin@plain%
81     \marginpar{\itxplainmarginfont{\ITX@itxs{#1}{#2}}}%
82     %\marginnote{\itxplainmarginfont{\ITX@itxs{#1}{#2}}}%
83   \fi%
84   \or%

...an acronym, ...
85   \if@itx@margin@acronym%
86     \marginpar{\itxacronymmarginfont{\ITX@itxs{#1}{#2}}}%
87   \fi%
88   \or%

...or a person's name.
89   \if@itx@margin@person%
90     \marginpar{\itxpersonmarginfont{\ITX@itxl{#1}{#2}}}%
91   \fi%
92   \fi%
93 }%

\ITX@used Value to flag a concept as used.
94 \newcommand*\ITX@used{@<>@<>@}%

\ITX@get
95 \newcommand*\ITX@get[2]{%
96   \ifx#1\relax%
97   \else%
98     \expandafter#2#1%
99   \fi%
100}%

\itxplainarea Significant-area definitions. When these counters change, the concepts concerned
\itxacronymarea will be typeset as first occurrences.
\itxpersonarea
\@itxarea 101 \newcommand*\itxplainarea{\thesubparagraph:\thepage}%
102 \newcommand*\itxacronymarea{\thechapter}%
103 \newcommand*\itxpersonarea{\thesubsubsection}%
104 \newcommand*\@itxarea[1]{%
105   \ifcase#1%
106     {\itxplainarea}%
107   \or%
108     {\itxacronymarea}%
109   \or%
110     {\itxpersonarea}%
111   \fi%
112}%

\itx@last@pos0 The default (empty) area definitions.
\itx@last@pos1 113 \def\itx@last@pos0{%
\itx@last@pos2 114 \def\itx@last@pos1{%
115 \def\itx@last@pos2{%

```

```

\ITX@itxs
116 \newcommand*\ITX@itxs[2]{%
117   \csname fnss@number#2\endcsname%
118 }%

\ITX@itxl
119 \newcommand*\ITX@itxl[2]{%
120   \csname fns1@number#2\endcsname%
121 }%

\itxs The syntax is \itxs{<type>}{<identity>}. A wrapper for \@itxs.
122 \newcommand*\itxs[2]{%
123   \texorpdfstring{\protect\@itxs{#1}{#2}}{#1}}%

\@itxs The syntax is \@itxs{<type>}{<identity>}. Typesets the concept referred to by
<identity> in its short form according to its <type>.
124 \newcommand*\@itxs[2]{%
125   \ifcase\number#1%

Plain Concept

126     \itxplainfollowfont{\ITX@itxs{#1}{#2}}%
127     \or%

Acronym

128     \itxacronymshortfont{\ITX@itxs{#1}{#2}}%
129     \or%

Person

130     \itxpersonlastfont{\ITX@itxl{#1}{#2}}%
131     \fi%
132   }%

\itxl The syntax is \itxl{<type>}{<identity>}. A wrapper for \@itxl.
133 \newcommand*\itxl{\protect\@itxl}%

\@itxl The syntax is \@itxl{<type>}{<identity>}. Typesets the concept referred to by
<identity> in its long form according to its <type>.
134 \newcommand*\@itxl[2]{%
135   %\ITX@itxl{#1}{#2}%
136   \ifcase\number#1%

Plain Concept Typeset the concept, ...

137     \itxplainfollowfont{\ITX@itxs{#1}{#2}}\nolinebreak %
138     \or%

```

Acronym Typeset the concept (note in-between margin label), ...

```
139      \itxacronymdefont{\ITX@itxl{#1}{#2}}%
140      \or%
```

Person Typeset the concept (note the in-between margin label), ...

```
141      \itxpersondefont{%
142      \itxpersonfirstfont{\ITX@itxs{#1}{#2}} %
143      %\nolinebreak[3] %
144      \itxpersonlastfont{\ITX@itxl{#1}{#2}}%
145      }%
146      \fi%
147      }%
```

\itxf The syntax is `\itxf{<type>}{<identity>}`. A wrapper for `\@itxf`.

```
148 \newcommand*{\itxf}[2]{%
149 \texorpdfstring{\protect\@itxf{#1}{#2}}{\ITX@itxl{#1}{#2} (#1)}}%
150 }%
```

\@itxf The syntax is `\@itxf{<type>}{<identity>}`. Typesets the concept referred to by *<identity>* in its *full form* according to its *<type>*.

```
151 \newcommand*{\@itxf}[2]{%
152 \ifcase\number#1%
```

Plain Concept Typeset margin-notes if applicable, ...

```
153      \@itxmarginlabel{#1}{#2}%

      ...typeset the concept, ...

154      \itxplaindefont{\ITX@itxs{#1}{#2}}\nolinebreak %
155      \or%
```

Acronym Typeset the concept (note in-between margin label), ...

```
156      \itxacronymdefont{%
157      \ITX@itxl{#1}{#2} %
158      %\nolinebreak[3] %

      ...typeset margin-notes if applicable, ...

159      \@itxmarginlabel{#1}{#2}%

      ...continue typesetting the concept.

160      \itxacronymdefshortfont{%
161      \itxacronymshortfont{(\ITX@itxs{#1}{#2})}}%
162      }%
163      \or%
```

Person Typeset the concept (note the in-between margin label), ...

```

164         \itxpersondefont{%
165         \itxpersonfirstfont{%
166             \ITX@itxs{#1}{#2}} %
167             %\nolinebreak[3] %

...typeset margin-notes if applicable, ...

168         \@itxmarginlabel{#1}{#2}%

...continue typesetting the concept.

169         \itxpersonlastfont{%
170         \ITX@itxl{#1}{#2}%
171         }%
172     }%
173 \fi%
```

Now, do the used/unused accounting.

```

174 \expandafter\ifx\csname itx@#2\endcsname\ITX@used%
175     %\relax%
176 \else%
177     \global\expandafter\let\csname itx@#2\endcsname\ITX@used%
178     %\ITX@addtoclearlist{#2}% MTR
179 \fi%
180 %\ITX@logged{#2} MTR
181 }%
```

\@itxrecordarea The syntax is `\@itxrecordarea{<type>}{<identity>}`. A macro used to update the current used/non-used status of each concept. This macro only use the *type* and *numeric id* of the concept.

```

182 \newcommand*{\@itxrecordarea}[2]{%

Record this area:

183 \edef\curr@pos{\@itxarea{#1}}%
184 %\PackageWarning{InTeX}{Current position for type "#1" is "\curr@pos"}%

Remember the last area where this concept (second argument) was used.

185 \edef\last@pos{\csname itx@last@pos#1@#2\endcsname}%
186 \ifx\curr@pos\last@pos%

We're still in the same area. Hence, we do nothing.

187 \else%

The area has changed.

188     \ITX@reset{#2}%
189 \fi%
190 \expandafter\xdef\csname itx@last@pos#1@#2\endcsname{\curr@pos}%
191 }%
```

`\@itx@init@nonbookmode` The syntax is `\@itx@init@nonbookmode`. This command is responsible for setting up miscellaneous aspects of the package when used in nonbook environments.

```
192 \newcommand\@itx@init@nonbookmode{%
193   \newcommand*\@itxtypeset}[3]{% Article-mode.
194     \ifx##3A%
195       %\PackageWarning{InTeX}{Typesetting format is automatic}%
196       \expandafter\ifx\csname itx@##2\endcsname\ITX@used%
```

The concept was last used in this area. Hence, it is typeset in its short form.

```
197       \itxs{##1}{##2}%
198     \else%
```

The concept has not yet been used, or it was last used in another area. Hence, it is typeset in its full form.

```
199       \itxf{##1}{##2}%
200     \fi%
201   \else%
```

Explicit selected typesetting format.

```
202     %\PackageWarning{InTeX}{Typesetting format = "##3"%}
203     \ifx##3S\itxs{##1}{##2}\fi%
204     \ifx##3L\itxl{##1}{##2}\fi%
205     \ifx##3F\itxf{##1}{##2}\fi%
206   \fi%
207 }%
208 }%
```

`\@itx@init@bookmode` The syntax is `\@itx@init@bookmode`. This command is responsible for setting up miscellaneous aspects of the package when used in book environments.

```
209 \newcommand\@itx@init@bookmode{%
210   \PackageInfo{InTeX}{Adjusting behavior to suite book/report document
211     classes\@gobble}%
212   \newcommand*\@itxtypeset}[3]{% Book/report-mode.
213     \ifx##3A%
214       %\PackageWarning{InTeX}{Typesetting format is automatic}%
```

Automatic typesetting.

```
215     \if@mainmatter%
216       \expandafter\ifx\csname itx@##2\endcsname\ITX@used%
```

The concept was last used in this area. Hence, it is typeset in its short form.

```
217       \itxs{##1}{##2}%
218     \else%
```

The concept has not yet been used, or it was last used in another area. Hence, it is typeset in its full form.

```
219       \itxf{##1}{##2}%
220     \fi%
221   \else%
```

Either in frontmatter or in backmatter.

```
222      \itxl{##1}{##2}%
223      \fi%
224      \else%
```

Explicit selected typesetting format.

```
225      %\PackageWarning{InTeX}{Typesetting format = "##3"}%
226      \ifx##3S\itxs{##1}{##2}\fi%
227      \ifx##3L\itxl{##1}{##2}\fi%
228      \ifx##3F\itxf{##1}{##2}\fi%
229      \fi%
230  }%
231 }
```

`\@itxtypeset` The syntax is `\@itxtypeset{<type>}{<identity>}{<format code>}`. This command is responsible for typesetting the $(\langle type \rangle, \langle identity \rangle)$ tuple.

If the $\langle identity \rangle$ was referred to in the frontmatter (part of books and reports), then a different set of rules should dictate the typesetting of the according concept. Therefore, first find out if the package is used in an article, which does not have any `\if@mainmatter` macro.

```
232 \newif\if@itx@bookmode%
233 \@itx@bookmodefalse%
```

Adjust the defaults so they make sense when used with the article document class.

```
234 \@ifclassloaded{article}{%
235   \PackageInfo{InTeX}{Adjusting behavior to suite the article document
236     class\@gobble}%
237   \@itx@init@nonbookmode%
238   \@itx@bookmodefalse%
239 }{}%
```

Adjust the defaults so they make sense when used with the beamer document class.

```
240 \@ifclassloaded{beamer}{%
241   \PackageInfo{InTeX}{Adjusting behavior to suite the beamer document
242     class\@gobble}%
243   \@itx@init@nonbookmode%
244   \@itx@bookmodefalse%
245   \renewcommand*\itxacronymarea{0}%
246   \@itx@margin@plainfalse%
247   \@itx@margin@acronymfalse%
248   \@itx@margin@personfalse%
249 }{}%
```

Assume that the current document class is one of the book or report classes, or another class with a definition of `\thechapter`.

```
250 \if@itx@bookmode%
251   \@itx@init@bookmode%
252 \fi%
```

`\@itxplain` The syntax is `\@itxplain{⟨type⟩}{⟨identity⟩}{⟨format code⟩}`. This command is responsible for keeping track of where the $(\langle type \rangle, \langle identity \rangle)$ tuple was last used and for typesetting it accordingly.

The $\langle format code \rangle$ can be one of

A for *automatic* selection of any of the following, explicit, format codes.

S for typesetting the entry in its *short* form.

L for typesetting the entry in its *long* form.

F for *full-form* typesetting.

```
253 \newcommand*{\@itxplain}[3]{%
```

First, update the “last used” status of the current concept so that it refers to the current area.

```
254 \@itxrecordarea{#1}{#2}%
```

Then, typeset the concept.

```
255 \@itxtypeset{#1}{#2}{#3}%
```

```
256 }%
```

```
257 % \end{macro}
```

```
258 %
```

```
259 % \begin{macro}{\@itxalias}
```

```
260 % Define the identity of the equivalent entry. Get the identity of the main
261 % index entry for which this is an alias. Keep the original
262 % definition as |\@orig|. Redefine the main entry (as in
263 % |\expandafter\gdef\csname fn@#1\endcsname{#2}{#2}|.) Now,
264 % typeset the alias by using the main index entry identity. Finally,
265 % reset the definition of the main entry. \textbf{|FIXME:|} This
266 % command is (probably) not used at the moment (2007-07-14) and is
267 % not doing what this paragraph states.
```

```
268 % \begin{macrocode}
```

```
269 \newcommand*{\@itxalias}[2]{%
```

```
270 \edef\@mainserial{\expandafter\@firstoftwo#2}%
```

```
271 \edef\@equivserial{\expandafter\@secondoftwo#2}%
```

Record usage of the main concept entry.

```
272 %\@itxrecordarea{#1}{\@mainserial}%
```

```
273 \@itxplain{#1}{\@equivserial}%
```

```
274 }%
```

`\@itx@fakeindex` If no index is to be generated, we still need some output for `mkintex` to work with. Hence, this command substitutes the `\index` command in the `index` package, and makes sure that an index entry with *page* equal to `\thepage` is written to the auxiliary file instead. However, the *page* value will then be used only for providing more detailed warnings about references to undefined concepts.

```
275 \newcommand{\@itx@fakeindex}[1]{%
```

```
276 \begingroup%
```

```
277 \edef\@tempa{%
```



```

278     \write\@auxout{%
279         \string\@writefile{raw}{%
280             \string\indexentry{#1}{\thepage}%
281         }%
282     }%
283 }%
284 \expandafter\endgroup\@tempa%
285 }%

```

`\co` The `\co` command is the only command the user should *need* to use. The syntax is `\co{<identity>}[<format code>]`, where `<identity>` refers to a concept and `<format code>` is an optional argument that can be used to force a particular kind of typesetting. The idea is that the `\co{}` command should be wrapped around every concept the user want to either typeset or index in a special and consistent way (or both). Note that `\co` is a wrapper for the `\@itx` command. Please see the definition of `\@itxplain` for a description of the available format codes. The default `<format code>` is A.

```

286 %\newcommand*\co{\protect\@itx}%
287 \newcommand*\co{\@itx}%
288 \newcommand*\coS[1]{\@itx[S]{#1}}%
289 \newcommand*\coL[1]{\@itx[L]{#1}}%
290 \newcommand*\coF[1]{\@itx[F]{#1}}%
291 %\DeclareRobustCommand*\co{\protect\@itx}%

```

`\@itx` The `\@itx` command works just as described for `\co` above. Hence, `#1` is the `<format code>`, and `#2` is the `<identity>`.

```

292 %\newcommand*\@itx[2][A]{%
293 \DeclareRobustCommand*\@itx[2][A]{%
294     \def\@tempa{#2}%

```

Handle, e.g., backslashes.

```

295     \edef\@tempb{\@nearverbatim\@tempa}%

```

If *LaTeX* should generate an index, simply use the `index` package to write the identifying index entry. The intricacies of how this concept should be indexed is handled externally by the `mkintex` program that is part of the *LaTeX* package.

```

296     \if@itx@index%
297         \index[raw]{#2}%
298     \else%
299         \@itx@fakeindex{\@tempb}%
300     \fi%

```

Define a new conditional, `found`, to signal whether the `<identity>` is found.

```

301     \newif\iffound%

```

Now, let the `co@type` counter loop through the values `[0, 1, 2]`.

```

302     \setcounter{co@type}{0}%
303     \loop\ifnum\theco@type<3%

```

Check to see if the $\langle identity \rangle$ is an (acronym or person) *alias* or a *main* entry. If it is an *alias*, there exists a variable named `fnne@ $\langle identity \rangle$` (note the extra ‘e’). However, if $\langle identity \rangle$ refers to a *main* entry, a variable named `fnn@ $\langle identity \rangle$` exists; that is, without the ‘e’ before the ‘@’. Also, it should be noted that *no alias can exist without a main entry with the same $\langle identity \rangle$* .

If an expansion of $\langle identity \rangle$ is found, typeset it accordingly and flag the finding by setting `\iffound`.

```
304 \expandafter\ifx\csname fn\number\theco@type e@\@tempb\endcsname\relax%
305 \expandafter\ifx\csname fn\number\theco@type @\@tempb\endcsname\relax%
```

The $\langle identity \rangle$ may refer to both a main entry *and* an alias entry (because of the requirement mentioned above). Do nothing. The reason the code is written in this way is to implement a preference to main entries over alias entries (see the order below).

```
306 % \PackageWarning{InTeX}{Main AND alias reference ‘#2’ occurred}%
307 % The reference refers to both main _and_ alias entries. Do
308 % nothing, this will be resolved through the |else|-cases
309 % immediately below.
310 \else%
```

The $\langle identity \rangle$ refers to a main-entry.

```
311 % \PackageInfo{InTeX}{Main reference ‘#2’ occurred}%
312 \edef\co@id{\csname fn\number\theco@type @\@tempb\endcsname}%
313 \@itxplain{\number\theco@type}{\co@id}{#1}%
314 \foundtrue%
315 \fi%
316 % \else%
```

The $\langle identity \rangle$ refers to an alias-entry.

```
317 % \PackageWarning{InTeX}{Alias reference ‘#2’ occurred}%
318 % XXX: It seems that this case never occurs.
319 % \edef\co@id{\csname fn\number\theco@type e@\@tempb\endcsname}%
320 % \@itxalias{\number\theco@type}{\co@id}%
321 % \foundtrue%
322 \fi%
```

Increase `co@type` by 1 and perform a new iteration of the loop. In other words, check if the reference ($\langle identity \rangle$) is referring to another kind of entry.

```
323 \stepcounter{co@type}%
324 \repeat%
```

If no expansion of $\langle identity \rangle$ could be found, warn the user. Furthermore, an in-document warning will be typeset if `@itx@warn@undef` is *true*.

```
325 \iffound%
326 % Do nothing.
327 \else%
328 \PackageWarning{InTeX}{Reference ‘#2’ to undefined concept}%
329 \if@itx@warn@undef%
330 \textbf{\itxundefcomment{#2}}%
331 \else%
```

```

332      #2%
333      \fi%
334 \fi%
335 }%

\personused
336 \newcommand*{\personused}[1]{%
337   \expandafter\ifx\csname pnused@#1\endcsname\PN@used%
338     \relax%
339   \else%
340     \global\expandafter\let\csname pnused@#1\endcsname\PN@used%
341     \global\let\PN@populated\PN@used%
342   \fi%
343 }%

\@itxdefineforms
344 \newcommand\@itxdefineforms[3]{%
345   \expandafter\gdef\csname fnss@number#1\endcsname{#2}%
346   \expandafter\gdef\csname fns1@number#1\endcsname{#3}%
347 }%

\@newentry The macros \new<type>, where <type> ∈ {acronym, concept, person} (as described
below), all call \@newentry with an additional first argument, namely the numeric
<type> identifier of the new entry. The syntax is

\@newentry{<type>}{<reference>}{<typeset 1>}{<typeset 2>},

where <reference> is the string later referred to as \co{<string>}, and <typeset 1>
and <typeset 2> define how the concept will be typeset where it was referred to
in the text. The exact meaning of <typeset 1> and <typeset 2> depends on what
<type> this entry has.

348 \newcommand\@newentry[4]{%
349   \def\@tempa{#2}%
350   \edef\@tempb{\@nearverbatim\@tempa}%
351   %
352   \stepcounter{co@serial}%
353   %\PackageWarning{init}{serial counter = \expandafter\theco@serial}%
354   \expandafter\xdef\csname fn\number#1@\@tempb\endcsname{%
355     \number\theco@serial}%
356   %\PackageWarning{init}{Def: \meaning\csname fn\number#1@\@tempb\endcsname}%
357   \@itxdefineforms{\theco@serial}{#3}{#4}%
358 }%

\newconcept
359 \newcommand*\newconcept[3]{%
360   \@newentry{0}{#1}{#2}{#3}%
361 }%

```

`\newacronym`

```
362 \newcommand*\newacronym[3]{%
363   \@newentry{1}{#1}{#2}{#3}%
364 }%
```

`\newperson` The syntax is

`\newperson{<reference>}{<short-form typeset>}{<full-form typeset>}`.

```
365 \newcommand*\newperson[3]{%
366   \@newentry{2}{#1}{#2}{#3}%
367 }%
```

`\@newentryequiv` The syntax for this command is

`\@newentryequiv{<type>}{<parent>}{<typeset>}{<reference>}{<full-form typeset>}`,

where $\langle type \rangle \in \{\text{acronym}, \text{concept}, \text{person}\}$, $\langle parent \rangle$ is the (reference) identity of the concept for which this is an equivalent, $\langle typeset \rangle$ defines how this equivalent should be typeset in text (short-form if it is an acronym), $\langle reference \rangle$ is the identity of this entry (referred to as `\co{<reference>}`), and $\langle full-form typeset \rangle$ defines how this concept should be typeset in the text in its full-form.

Note that the macros `\new<type>equiv`, where $\langle type \rangle \in \{\text{acronym}, \text{concept}, \text{person}\}$, all are wrappers for this command.

```
368 \newcommand*\@newentryequiv[5]{%
369   \newif\iffound%
370   \def\@tempa{#2}%
371   \edef\@tempb{\@nearverbatim\@tempa}%
372   \expandafter\ifx\csname fn\number#1 \@tempb\endcsname\relax%
    Nothing is done if \csname fn\number#1 \@tempb\endcsname is not defined
    here, but notice that the default value of found is false.
373   \else%
374     \foundtrue%
375     \edef\co@id{\csname fn\number#1\@tempb\endcsname}%
376     %\PackageWarning{init}{Found ‘\@tempb’ (type=\number#1, serial=\co@id)}%
```

Store the *short* and the *long* versions of the alias, in that order.

```
377   \stepcounter{co@serial}%
378   \@itxdefineforms{\theco@serial}{#3}{#5}%
```

Store the numeric identity of the concept alias.

```
379   \def\@tempa{#4}%
380   \edef\@tempb{\@nearverbatim\@tempa}%
381   \expandafter\xdef\csname fn\number#1 e\@tempb\endcsname{%
382     {\co@id}{\theco@serial}}%
383   \fi%
384   \iffound%
385   \else%
386     %\PackageWarning{InTeX}{Can’t find ‘#2’ for sub-concept ‘#3’}%
387   \fi%
388 }%
```

`\newconceptequiv` Wrappers for the `\@newentryequiv` command, where each wrapper specifies the entry's *<type>* through the first argument to `\@newentryequiv`.

`\newacronyमेquiv`

`\newpersonequiv`

```

389 \newcommand*\newconceptequiv[4]{%
390   \@newentryequiv{0}{#1}{#2}{#3}{#4}%
391 }%
392 \newcommand*\newacronyमेquiv[4]{%
393   \@newentryequiv{1}{#1}{#2}{#3}{#4}%
394 }%
395 \newcommand*\newpersonequiv[4]{%
396   \@newentryequiv{2}{#1}{#2}{#3}{#4}%
397 }%
```

5.6 The Internal *InT_EX* File

`\InTeX` After the first run of `MakeInTEX`, the file `\jobname.ito` will contain the different concept definitions. The `.ito` file is loaded at the beginning of the document.

```

398 \AtBeginDocument{\input{\jobname.ito}}%
```

Index

Bokmålsordboka, 3

carnivore, 3

 canine, 3

 feline, 3

compound, 3

 —word analyzer, 3

 non-deverbal verb —, 3

 Norwegian —word analysis, 3

concept, 2, 6

 acronym —, 2

 person —, 2

 plain —, 2

 sub—, 2

dihydrogen monoxide, 2, 3

 — reserve, 2

H₂O, *see* dihydrogen monoxide

 — reserve, *see* —, — reserve

ID, *see* identity

identity, 3

index, 4

LaTeX logo, 6

m^ea_t eater, *see* carnivore

Presley, Elvis Aaron, 4

Python, 2

Semantic Information for Multifunctional Plurilingual Lexica, 3

 Swedish —, 3

SIMPLE, *see* *Semantic Information for Multifunctional Plurilingual Lexica*

 Swedish —, *see* —, Swedish —

dog (*explicitly sorted*), 4

vertex, 4

water, *see* dihydrogen monoxide