The *In*TeX Package[*]

Martin Thorsen Ranang
`mtr@ranang.org`

2013/03/12

1 Introduction

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

`intex` `\co`

The really short usage description is that in order to use the package, insert `\usepackage{intex}` at the beginning of your LaTeX 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 LaTeX 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 LaTeX 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` `index`

Already, packages exist that provide functionality that eases the acronym[1] and indexing[2] 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 occurences—perhaps typeset several chapters apart—to be indexed as being the same concept.

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

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

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

The *In*TeX 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.

Python

3 Usage

How you can use *In*TeX should be clearer after examining some examples. The central idea in *In*TeX 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.

concept
acronym
person
concepts
acronym
person
plain
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" where also defined as plain. However, these concepts are defined as *sub-concepts* of "concept" and should be indexed accordingly.

sub-concepts

3.1 Package Options

noindex : Whether *In*TeX should generate an index or not. (*Default: true*).

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

nomargin⟨*type*⟩ : Tell *In*TeX 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

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)*

```
\makeatletter
\@itx@margin@acronymfalse%
\makeatother
It is easy to refer to (and thus index)
acronyms, like \co{H2O}.  And
sub-concepts, like \co{H2O reserve}
```

[3] Python 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 *(ID)*”

```
\makeatletter
\@itx@margin@acronymfalse%
\makeatother
We could talk about multiple \co{IDs},
or a single \co{ID}.  The following
table shows explicitly defined
formatting:\\

\begin{tabular}{@{}ll@{}}
  \toprule
  Format & Expansion \\
  \midrule
  forced long & ‘‘\coL{ID}’’\\
  short & ‘‘\coS{ID}’’\\
  full & ‘‘\coF{ID}’’ \\
  \bottomrule
\end{tabular}
```

The *In*TeX 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.

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

compound-word analyzer

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

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 *In*TeX-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.

```
The word \co{carnivore} means
\co{meat eater}.  There are both
\co{feline} and \co{canine}
\co{meat eaters} in the animal
kingdom.
```

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.

```
\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 *In*TeX-file. The same goes for *vertex* and *vertices*, and *dog* and *dogs*.

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

the King
Presley

The *In*TeX 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:

```
% -*- 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`.

```
%
% $Id$
%
% name=main
% default_inflection=singular
%
% Copyright (C) 2005--2007 by Martin Thorsen Ranang
%

```

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:

```
% In general, to refer to the full-form of a concept, use the \cof
% macro.
%
% *ACRONYMS*
AI	⇥Artificial Intelligence
```

```
  AIC   ⇥- Complete

H2O@H$_{2}$O   ⇥dihydrogen monoxide
  - reserve

ID   ⇥identity

SIMPLE   ⇥\textit{Semantic Information for Multifunctional Plurilingual Lexica}   ⇥:sort_as=Semant
  Swedish -   ⇥\textit{Swedish -}

synset   ⇥synonym set

% *PEOPLE*
EAP   ⇥Presley, Elvis Aaron
EAPK   ⇥King, the --> EAP
JFK   ⇥Kennedy, John Fitzgerald
MLK   ⇥King, Martin Luther
Madonna   ⇥Madonna
VWvG   ⇥van Gogh, Vincent Willem

% *CONCEPTS*
Bokmålsordboka@\textit{Bokmålsordboka}   ⇥:sort_as=Bokmålsordboka

carnivore
  feline
    domestic cat
    tiger
  canine
    Gray Wolf
    domestic dog
compound@com\\\-pound
  -\-word analyzer
  non\-deverbal verb -
  Norwegian -\-word analysis
concept
  acronym (-)
  person (-)
  plain (-)
  sub\--

dog   ⇥:sort_as=sog me bold :comment=\textit{(explicitly sorted)}

idea
index

(meat eater)@{m$^{\textrm{e}}$a$_{\textrm{t}}$ eater} --> carnivore

synonymy
synonym --> synonymy
  direct -
  indirect -

(InTeX logo)@{{\InTeX\ logo}}

Python

TeXnician@{\TeX}nician
  -'s tool

vertex

water --> H2O

% Local Variables:
% mode: latex
% TeX-master: t
% ispell-local-dictionary: "american"
% mode: flyspell
% End:
```

3.3 Compilation

mkintex

concept

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

1. latex ⟨*name*⟩.tex
2. mkintex ⟨*name*⟩ ⟨*name*⟩.itx -o ⟨*name*⟩.rix [-a acronyms.tex -p persons.tex]
3. makeindex ⟨*name*⟩
4. makeindex -o ⟨*name*⟩.rid ⟨*name*⟩.rix
5. latex ⟨*name*⟩.tex

4 Macros

\InTeX

*In*TEX logo

This is simply a macro for typesetting the *InTEX logo*.

5 Implementation

After the customary identification,

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

we continue by defining the package options.

5.1 Package Options

noindex
\if@itx@index

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

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

nowarnundef
\if@itx@nowarnundef

The conditional `\if@itx@nowarnundef` controls whether *In*TeX 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.

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

nomarginplain
nomarginacronym
nomarginperson
\if@itx@margin@plain
\if@itx@margin@acronym
\if@itx@margin@person

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.

```
\newif\if@itx@margin@plain%
\newif\if@itx@margin@acronym%
\newif\if@itx@margin@person%
\@itx@margin@plaintrue%
\@itx@margin@acronymtrue%
\@itx@margin@persontrue%
\DeclareOption{nomarginplain}{\@itx@margin@plainfalse}%
\DeclareOption{nomarginacronym}{\@itx@margin@acronymfalse}%
\DeclareOption{nomarginperson}{\@itx@margin@personfalse}%
```

Next, process the options.

```
\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.

```
\if@itx@index%
  \RequirePackage{index}%
  \makeindex%
  \newindex{raw}{rix}{rid}{Index}%
\else%
  \def\@nearverbatim{\expandafter\strip@prefix\meaning}%
\fi%
```

marginnote

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

```
\if@itx@margin@plain%
  \RequirePackage{marginnote}[2006/10/26]%
\fi%
\if@itx@margin@acronym%
  \RequirePackage{marginnote}[2006/10/26]%
\fi%
\if@itx@margin@person%
  \RequirePackage{marginnote}[2006/10/26]%
\fi%
```

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

acronym
ifthen

Anyhow, require the acronym and the ifthen packages to be loaded.

```
\RequirePackage{acronym}[2008/05/28]%
\RequirePackage{ifthen}%
```

5.3 The *In*TeX Logo

\InTeX

Define a TeX-ish logo for this package.

```
\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
\itxplainfollowfont
\itxplainmarginfont

These commands are used to typeset plain concepts.

```
\newcommand\itxplaindeffont[1]{\emph{#1}}%
\newcommand\itxplainfollowfont[1]{#1}%
\newcounter{itxpl}%
\newcommand\@itxbasemarginfont[1]{%
  \stepcounter{itxpl}%
  \ifthenelse{\isodd{\pageref{itxpl-\theitxpl}}}{%
    \raggedright\hspace{0pt}\footnotesize\textsf{#1}% odd
  }{%
    \raggedleft\hspace{0pt}\footnotesize\textsf{#1}% even
  }%
  \label{itxpl-\theitxpl}%
}%
\newcommand\itxplainmarginfont[1]{%
  \@itxbasemarginfont{#1}%
}%
```

\itxacronymdeffont
\itxacronymdefshortfont
\itxacronymshortfont
\itxacronymmarginfont

For acronyms:

```
\newcommand{\itxacronymdeffont}[1]{#1}%
\newcommand{\itxacronymdefshortfont}[1]{\emph{#1}}%
\newcommand{\itxacronymshortfont}[1]{#1}%
\newcommand{\itxacronymmarginfont}[1]{%
```

```
  \@itxbasemarginfont{#1}%
  %\raggedleft\hspace{0pt}\footnotesize\textsf{#1}%
}%
```

\itxpersondeffont
\itxpersonfirstfont
\itxpersonlastfont
\itxpersonmarginfont

For persons:

```
\newcommand{\itxpersondeffont}[1]{\emph{#1}}%
\newcommand{\itxpersonfirstfont}[1]{#1}%
\newcommand{\itxpersonlastfont}[1]{#1}%
\newcommand{\itxpersonmarginfont}[1]{%
  \@itxbasemarginfont{#1}%
  %\raggedleft\hspace{0pt}\footnotesize\textsf{#1}%
}%
```

5.5 The (Low-Level) Clockwork of the Package

co@serial

First, define a counter that is used to enumerate new concept definitions.

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

co@equiv@serial

First, define a counter that is used to enumerate new concept definitions.

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

co@type

First, define a counter that is used to enumerate new concept definitions.

```
\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.

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

Define a couple of convenience macros.

```
\long\def\@firstofthree#1#2#3{#1}%
\long\def\@secondofthree#1#2#3{#2}%
%\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.

```
\def\ITX@reset#1{%
  \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.

```
\newcommand*\@itxmarginlabel[2]{%
  \hspace{0pt}%
```

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. . .

```
  \ifcase#1%
```

. . . plain concept, . . .

```
    \if@itx@margin@plain%
      \marginpar{\itxplainmarginfont{\ITX@itxs{#1}{#2}}}%
      %\marginnote{\itxplainmarginfont{\ITX@itxs{#1}{#2}}}%
    \fi%
  \or%
```

. . . an acronym, . . .

```
    \if@itx@margin@acronym%
      \marginpar{\itxacronymmarginfont{\ITX@itxs{#1}{#2}}}%
    \fi%
  \or%
```

. . . or a person's name.

```
    \if@itx@margin@person%
      \marginpar{\itxpersonmarginfont{\ITX@itxl{#1}{#2}}}%
    \fi%
  \fi%
}%
```

\ITX@used

Value to flag a concept as used.

```
\newcommand*\ITX@used{@<>@<>@}%
```

\ITX@get

```
\newcommand*\ITX@get[2]{%
  \ifx#1\relax%
  \else%
    \expandafter#2#1%
  \fi%
}%
```

\itxplainarea
\itxacronymarea
\itxpersonarea
\@itxarea

Significant-area definitions. When these counters change, the concepts concerned will be typeset as first occurrences.

```
\newcommand*\itxplainarea{\thesubparagraph:\thepage}%
\newcommand*\itxacronymarea{\thechapter}%
\newcommand*\itxpersonarea{\thesubsubsection}%
\newcommand*\@itxarea[1]{%
  \ifcase#1%
    {\itxplainarea}%
  \or%
    {\itxacronymarea}%
  \or%
    {\itxpersonarea}%
  \fi%
}%
```

\itx@last@pos0
\itx@last@pos1
\itx@last@pos2

The default (empty) area definitions.

```
\def\itx@last@pos0{}%
\def\itx@last@pos1{}%
\def\itx@last@pos2{}%
```

\ITX@itxs

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

\ITX@itxl

```
\newcommand*\ITX@itxl[2]{%
  \csname fnsl@\number#2\endcsname%
}%
```

\itxs The syntax is `\itxs{`⟨*type*⟩`}{`⟨*identity*⟩`}`. A wrapper for `\@itxs`.

```
\newcommand*{\itxs}[2]{%
  \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*⟩.

```
\newcommand*{\@itxs}[2]{%
  \ifcase\number#1%
```

Plain Concept

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

Acronym

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

Person

```
         \itxpersonlastfont{\ITX@itxl{#1}{#2}}%
       \fi%
     }%
```

\itxl The syntax is `\itxl{`⟨*type*⟩`}{`⟨*identity*⟩`}`. A wrapper for `\@itxl`.

```
\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*⟩.

```
\newcommand*{\@itxl}[2]{%
  %\ITX@itxl{#1}{#2}%
  \ifcase\number#1%
```

Plain Concept Typeset the concept, . . .

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

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

```
         \itxacronymdeffont{\ITX@itxl{#1}{#2}}%
       \or%
```

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

```
         \itxpersondeffont{%
           \itxpersonfirstfont{\ITX@itxs{#1}{#2}} %
           %\nolinebreak[3] %
           \itxpersonlastfont{\ITX@itxl{#1}{#2}}%
         }%
       \fi%
     }%
```

\itxf The syntax is \itxf{⟨*type*⟩}{⟨*identity*⟩}. A wrapper for \@itxf.

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

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

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

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

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

. . . typeset the concept, . . .

```
         \itxplaindeffont{\ITX@itxs{#1}{#2}}\nolinebreak %
       \or%
```

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

```
         \itxacronymdeffont{%
           \ITX@itxl{#1}{#2} %
           %\nolinebreak[3] %
```

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

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

. . . continue typesetting the concept.

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

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

```
         \itxpersondeffont{%
           \itxpersonfirstfont{%
             \ITX@itxs{#1}{#2}} %
           %\nolinebreak[3] %
```

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

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

. . . continue typesetting the concept.

```
           \itxpersonlastfont{%
             \ITX@itxl{#1}{#2}%
           }%
         }%
       \fi%
```

Now, do the used/unused accounting.

```
  \expandafter\ifx\csname itx@#2\endcsname\ITX@used%
    %\relax%
  \else%
    \global\expandafter\let\csname itx@#2\endcsname\ITX@used%
    %\ITX@addtoclearlist{#2}% MTR
  \fi%
  %\ITX@logged{#2} MTR
}%
```

\@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.

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

Record this area:

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

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

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

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

```
  \else%
```

The area has changed.

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

\@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.

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

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

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

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

```
        \itxf{##1}{##2}%
      \fi%
    \else%
```

Explicit selected typesetting format.

```
      %\PackageWarning{InTeX}{Typesetting format = "##3"}%
      \ifx##3S\itxs{##1}{##2}\fi%
      \ifx##3L\itxl{##1}{##2}\fi%
      \ifx##3F\itxf{##1}{##2}\fi%
    \fi%
  }%
}%
```

\@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.

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

Automatic typesetting.

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

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

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

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

```
          \itxf{##1}{##2}%
        \fi%
      \else%
```

Either in frontmatter or in backmatter.

```
    \itxl{##1}{##2}%
   \fi%
  \else%
```

Explicit selected typesetting format.

```
   %\PackageWarning{InTeX}{Typesetting format = "##3"}%
   \ifx##3S\itxs{##1}{##2}\fi%
   \ifx##3L\itxl{##1}{##2}\fi%
   \ifx##3F\itxf{##1}{##2}\fi%
  \fi%
 }%
}
```

`\@itxtypeset` The syntax is `\@itxtypeset{`⟨*type*⟩`}{`⟨*identity*⟩`}{`⟨*format code*⟩`}`. This command is responsible for typesetting the (⟨*type*⟩, ⟨*identity*⟩) tuple.

If the ⟨*identity*⟩ 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.

```
\newif\if@itx@bookmode%
\@itx@bookmodetrue%
```

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

```
\@ifclassloaded{article}{%
 \PackageInfo{InTeX}{Adjusting behavior to suite the article document
  class\@gobble}%
 \@itx@init@nonbookmode%
 \@itx@bookmodefalse%
}{}%
```

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

```
\@ifclassloaded{beamer}{%
 \PackageInfo{InTeX}{Adjusting behavior to suite the beamer document
  class\@gobble}%
 \@itx@init@nonbookmode%
 \@itx@bookmodefalse%
 \renewcommand*\itxacronymarea{0}%
 \@itx@margin@plainfalse%
 \@itx@margin@acronymfalse%
 \@itx@margin@personfalse%
}{}%
```

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

```
\if@itx@bookmode%
 \@itx@init@bookmode%
\fi%
```

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

The ⟨*format code*⟩ 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.

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

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

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

Then, typeset the concept.

```
  \@itxtypeset{#1}{#2}{#3}%
}%
% \end{macro}
%
% \begin{macro}{\@itxalias}
%   Define the identity of the equivalent entry.  Get the identity of the main
%   index entry for which this is an alias.  Keep the original
%   definition as |\@orig|.  Redefine the main entry (as in
%   |\expandafter\gdef\csname fn@#1\endcsname{{#2}{#2}}|.)  Now,
%   typeset the alias by using the main index entry identity.  Finally,
%   reset the definition of the main entry.  \textbf{|FIXME:|} This
%   command is (probably) not used at the moment (2007-07-14) and is
%   not doing what this paragraph states.
%    \begin{macrocode}
\newcommand*{\@itxalias}[2]{%
  \edef\@mainserial{\expandafter\@firstoftwo#2}%
  \edef\@equivserial{\expandafter\@secondoftwo#2}%
```

Record usage of the main concept entry.

```
  %\@itxrecordarea{#1}{\@mainserial}%
  \@itxplain{#1}{\@equivserial}%
}%
```

`\@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.

```
\newcommand{\@itx@fakeindex}[1]{%
  \begingroup%
    \edef\@tempa{%
```

```
      \write\@auxout{%
        \string\@writefile{raw}{%
          \string\indexentry{#1}{\thepage}%
        }%
      }%
    }%
  \expandafter\endgroup\@tempa%
}%
```

`\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`.

```
%\newcommand*{\co}{\protect\@itx}%
\newcommand{\co}{\@itx}%
\newcommand{\coS}[1]{\@itx[S]{#1}}%
\newcommand{\coL}[1]{\@itx[L]{#1}}%
\newcommand{\coF}[1]{\@itx[F]{#1}}%
%\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*⟩.

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

Handle, e.g., backslashes.

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

If *In*TeX 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 *In*TeX package.

```
  \if@itx@index%
    \index[raw]{#2}%
  \else%
    \@itx@fakeindex{\@tempb}%
  \fi%
```

Define a new conditional, `found`, to signal whether the ⟨*identity*⟩ is found.

```
  \newif\iffound%
```

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

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

Check to see if the ⟨*identity*⟩ is an (acronym or person) *alias* or a *main* entry. If it is an *alias*, there exists a variable named `fn`*n*`e@`⟨*identity*⟩ (note the extra '`e`'). However, if ⟨*identity*⟩ refers to a *main* entry, a variable named `fn`*n*`@`⟨*identity*⟩ exists; that is, without the '`e`' before the '`@`'. Also, it should be noted that *no alias can exists without a main entry with the same* ⟨*identity*⟩.

If an expansion of ⟨*identity*⟩ is found, typeset it accordingly and flag the finding by setting `\iffound`.

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

The ⟨*identity*⟩ 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).

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

The ⟨*identity*⟩ refers to a main-entry.

```
    %\PackageInfo{InTeX}{Main reference '#2' occurred}%
    \edef\co@id{\csname fn\number\theco@type @\@tempb\endcsname}%
    \@itxplain{\number\theco@type}{\co@id}{#1}%
    \foundtrue%
   \fi%
%  \else%
```

The ⟨*identity*⟩ refers to an alias-entry.

```
%    \PackageWarning{InTeX}{Alias reference '#2' occurred}%
%    %  XXX: It seems that this case never occurrs.
%    \edef\co@id{\csname fn\number\theco@type e@\@tempb\endcsname}%
%    \@itxalias{\number\theco@type}{\co@id}%
%    \foundtrue%
  \fi%
```

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

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

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

```
 \iffound%
% Do nothing.
 \else%
  \PackageWarning{InTeX}{Reference '#2' to undefined concept}%
  \if@itx@warn@undef%
   \textbf{\itxundefcomment{#2}}%
  \else%
```

```
      #2%
    \fi%
  \fi%
}%
```

\personused

```
\newcommand*{\personused}[1]{%
  \expandafter\ifx\csname pnused@#1\endcsname\PN@used%
    \relax%
  \else%
    \global\expandafter\let\csname pnused@#1\endcsname\PN@used%
    \global\let\PN@populated\PN@used%
  \fi%
}%
```

\@itxdefineforms

```
\newcommand\@itxdefineforms[3]{%
  \expandafter\gdef\csname fnss@\number#1\endcsname{#2}%
  \expandafter\gdef\csname fnsl@\number#1\endcsname{#3}%
}%
```

\@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 refered to as `\co{`⟨*string*⟩`}`, and ⟨*typeset 1*⟩ and ⟨*typeset 2*⟩ define how the concept will be typeset where it was refered to in the text. The exact meaning of ⟨*typeset 1*⟩ and ⟨*typeset 2*⟩ depends on what ⟨*type*⟩ this entry has.

```
\newcommand\@newentry[4]{%
  \def\@tempa{#2}%
  \edef\@tempb{\@nearverbatim\@tempa}%
  %
  \stepcounter{co@serial}%
  %\PackageWarning{init}{serial counter = \expandafter\theco@serial}%
  \expandafter\xdef\csname fn\number#1@\@tempb\endcsname{%
    \number\theco@serial}%
  %\PackageWarning{init}{Def: \meaning\csname fn\number#1@\@tempb\endcsname}%
  \@itxdefineforms{\theco@serial}{#3}{#4}%
}%
```

\newconcept

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

\newacronym

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

\newperson The syntax is

`\newperson{`⟨*reference*⟩`}{`⟨*short-form typeset*⟩`}{`⟨*full-form typeset*⟩`}`.

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

\@newentryequiv The syntax for this command is

`\@newentryequiv{`⟨*type*⟩`}{`⟨*parent*⟩`}{`⟨*typeset*⟩`}{`⟨*reference*⟩`}{`⟨*full-form typeset*⟩`}`,

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

Note that the macros `\new`⟨*type*⟩`equiv`, where ⟨*type*⟩ ∈ {`acronym`, `concept`, `person`}, all are wrappers for this command.

```
\newcommand*\@newentryequiv[5]{%
  \newif\iffound%
  \def\@tempa{#2}%
  \edef\@tempb{\@nearverbatim\@tempa}%
  \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*.

```
  \else%
    \foundtrue%
    \edef\co@id{\csname fn\number#1@\@tempb\endcsname}%
    %\PackageWarning{init}{Found `\@tempb' (type=\number#1, serial=\co@id)}%
```

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

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

Store the numeric identity of the concept alias.

```
    \def\@tempa{#4}%
    \edef\@tempb{\@nearverbatim\@tempa}%
    \expandafter\xdef\csname fn\number#1 e@\@tempb\endcsname{%
      {\co@id}{\theco@serial}}%
  \fi%
  \iffound%
  \else%
    %\PackageWarning{InTeX}{Can't find `#2' for sub-concept `#3'}%
  \fi%
}%
```

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

```
\newcommand*\newconceptequiv[4]{%
  \@newentryequiv{0}{#1}{#2}{#3}{#4}%
}%
\newcommand*\newacronymequiv[4]{%
  \@newentryequiv{1}{#1}{#2}{#3}{#4}%
}%
\newcommand*\newpersonequiv[4]{%
  \@newentryequiv{2}{#1}{#2}{#3}{#4}%
}%
```

5.6 The Internal *In*TeX File

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

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

Index