

Extended references to items for enumitem package

FC

2010/12/10 – version 1.2

Abstract

`enumitem-zref`, as a companion package for `enumitem`¹, extends the references mechanism for lists of items and allows hyperlinks to any items in lists formatted by `enumitem`. Hyper-references to lists seems not always necessary, but their importance rise much when a document contains lists that can stretch out on several pages.

Primarily intended for `enumerate` environments (*ie.* numbered lists), `enumitem-zref` can also reference any item in `description` or `itemize` environments: restrictions can be set with the `package options`.

A `natural reference scheme` for referencing items anywhere in the document is provided with three modalities (the `commaref` scheme, the `refitem` scheme and the `user` scheme). The references can be typeset differently with the command `\refitem`, its options and eventually arbitrary text.

Additionally, `enumitem-zref` allows to typeset greek-enumerated lists with the `greek` package option, which uses code from packages `alphalph`² and `engrec`³.

References to items in external documents compiled with `enumitem-zref` is also possible with pdfTeX⁴.

`enumitem-zref` requires and is based on the packages `enumitem` by Javier Bezos and `zref`⁵ by Heiko Oberdiek. It works with an ϵ -TeX distribution of L^AT_EX.

Contents

Summary of the properties for \refitem	2	4.2 The package options	9
1 Forewords	3	4.3 Some constants	10
1.1 Numbered lists, not numbered lists and hyperref	3	4.4 Properties for the zref list: zen@list	10
1.2 A “natural” reference scheme for lists	3	4.5 Normalisation of references and item labels	11
1.2.1 The <code>commaref</code> scheme		4.6 Keys <code>name</code> and <code>name*</code> are added to enu- mitem environments	12
1.2.2 The <code>refitem</code> scheme		4.7 Overloading <code>\makelabel</code>	13
1.2.3 The <code>user</code> scheme: with the <code>\label</code> command		4.8 Creating references	14
2 User Interface	5	4.9 Extracting zref properties	15
2.1 Naming lists	5	4.10 Referencing the items: <code>\refitem</code>	15
2.2 Referencing the items : <code>\refitem</code> .	5	4.11 Undefined references and duplicate labels management	18
2.3 The package options	6	4.12 <code>\zeninfo</code> (If you are lost...)	18
2.4 If your are lost <code>\zeninfo</code> is your friend	7	4.13 Create the references At Begin Document for the <code>refitem</code> and the <code>commaref</code> schemes	19
3 Flow chart of expansion: How does it work ?	8	5 References	21
4 IMPLEMENTATION	9	6 History	21
4.1 Identification and requirements	9	[2010/12/10 v1.2]	21
		[2010/12/02 v1.1]	21
		7 Index	21

1. enumitem: CTAN:help/Catalogue/entries/enumitem.html

2. alphalph: CTAN:help/Catalogue/entries/alphalph.html

3. engrec: CTAN:help/Catalogue/entries/engrec.html

4. This is not xr but “pure” PDF.

5. zref: CTAN:help/Catalogue/entries/zref.html

This documentation is produced with the DocStrip utility.

→ To get the package, run: `etex enumitem-zref.dtx`

→ To get the documentation run (thrice): `pdflatex enumitem-zref.dtx`

To get the index, run: `makeindex -s gind.ist enumitem-zref.idx`

The .dtx file is embedded into this pdf file thank to embedfile by H. Oberdiek.

Summary of the properties for \refitem

Terms definitions

`\label{user}` This is a user *label*.
`\item[user]` This is a user *item*.
`\item` This is an *automatic* item.



User items are labeled, hence expanded (with `\protect`).

The `\label` command can be put inside the optional argument of `\item`:

`\item[text \label{label of the item}]`

or in the paragraph that follows.

Package `hyperref`'s `[verbose]` option reports all inserted anchor names in the `.log` file.

Syntax for \refitem

`\refitem {reference} [properties to display]`

Optional text containing properties
 Default is `\ref`

reference with either: $\left\{ \begin{array}{ll} \text{The } \textit{refitem} \text{ scheme:} & \text{listname}\{\text{label}\}\{\text{label}\}\{\text{label}\}\dots \\ \text{The } \textit{commaref} \text{ scheme:} & \text{listname,index,index,index},\dots \\ \text{The } \textit{user} \text{ scheme:} & \text{reference to a } \backslash\text{label} \text{ command} \end{array} \right.$

The order in which the properties are given is related to the depth of the list:

[<code>\ref</code>	<code>\ref</code>	<code>\ref</code>]	1. One
	↑	↑	↑		i) sub item
	refers to the second ancestor <i>Here an empty string</i>	refers to the parent 1.	refers to the item ii)		ii) sub item
					2. Two

Property	Description	Examples
<code>\label</code>	The item label: automatic or user defined by <code>\item[...]</code>	II) or •
<code>\label*</code>	For automatic labels in <code>enumerate</code> lists: the label without any text around (useful to “concatenate” labels). Otherwise, the same as <code>\label</code> .	II or •
<code>\ref</code>	If the <code>ref</code> key of <code>enumitem</code> has been set, the “evaluation” of the <code>ref</code> key. Otherwise, the same as <code>\label</code>	II) or •
<code>\ref*</code>	The same as <code>\label*</code>	II or •
<code>\listname</code>	The name given to the list with the key <code>name</code> .	
<code>\depth</code>	The list depth.	
<code>\index</code>	The index of the item in the list. Every item is counted.	
<code>\listctr</code>	The value of <code>\@listctr</code> for the item – only inside <code>enumerate</code> lists. May be different from <code>\index</code> if the list contains user <i>items</i> .	
<code>\type</code>	The type of the list: either <code>enumerate</code> or <code>itemize</code> or <code>description</code> .	
<code>\currentlabel</code>	L ^A T _E X's <code>\@currentlabel</code>	
<code>\anchor</code>	The name of the anchor set for the item.	
<code>\currentlabelname</code>	<code>hyperref</code> 's (or <code>nameref</code> 's) <code>\@currentlabelname</code>	
<code>\refitem</code>	The list of references with the <i>refitem</i> scheme for the item.	~\zeninfo
<code>\commaref</code>	The list of references with the <i>commaref</i> scheme for the item.	~\zeninfo
<code>\default</code>	The same as <code>\ref</code>	

1 Forewords

1.1 Numbered lists, not numbered lists and hyperref

L^AT_EX defines two “classes” of lists of items:

- **enumerate** lists which are *numbered* (*ie.* a counter gives an index to each item in the list),
- **description** and **itemize** lists which are *not numbered* (no counter, thus no index).

When hyperref is used in *numbered lists*, each `\item` gives hyperref the opportunity to define an anchor, whose name is chosen considering the current value of the *index*.

Different cases have to be considered:

- 1) We are in a *numbered* list (*ie.* **enumerate**):
 - i) if the item is automatic (`\item` without option), hyperref puts an anchor named `Item.IDX`.
 - ii) if the item is user defined (`\item[user item]`), hyperref does not put an anchor.
- 2) We are not in a *numbered* list (*ie.* **description** or **itemize**):

hyperref does not put an anchor.

Moreover, when hyperref is loaded, the expansion of the `\label` command leads to the expansion of the item label (with `\protect`) for it is written to the `.aux` file:

- in the case 1 i)
- and also in the case 1 ii) if:
 - the list is a **description** list, and
 - package **nameref** is loaded, and
 - the `\label` command appears inside the **user item** text.

I was guardedly affraid about those rules when I understood them, as you may be...

enumitem-zref adds an anchor in the cases 1 ii) and 2). The item label text is always expanded (with `\protect` and `babel` protection).

1.2 A “natural” reference scheme for lists

enumitem-zref provides **3 ways** to reference a specific item in a list. The first 2 can be used to reference items in lists of external documents as well.

References are made with the command: `\refitem`, but `\hyperref` can be used as well.

`\subitem` and `\subsubitem` provided in some classes cannot be referenced with **enumitem-zref**.

1.2.1 The *commaref* scheme

Assuming you gave a name to your **enumerate**, **description** or **itemize** list, and let’s say this name is ‘**myList**’, each item is given an index. Any item is indexed, whether it’s printed by the mean of a user item or an an automatic item. Then:

- this is the item **myList,1**
- this is the item **myList,2**
 - this is the item **myList,2,1**
 - this is the item **myList,2,2**
- user-defined** this is the item **myList,2,3**

Reference to the third item can be made with:

<code>\refitem{myList,2,1}</code>	this prints: 1 for it is the first item in the inner list.
<code>\refitem{myList,2,1}[\ref.\ref]</code>	prints: 2.1.
<code>\refitem{myList,2,1}[\label\ \label]</code>	prints: • –.

Nothing simplest, no more comment...

1.2.2 The *refitem* scheme

- A) this is the item `myList{A}`.
- B) this is the item `myList{B}` (as you'd have expected...)
- i) this is the item `myList{B}{i}`.
and is *also* `innerlist{i}` if this nested list has been named 'innerlist'.
 - ii) this is the item `myList{B}{ii}` (or `innerlist{ii}`).
 - α this is the item `myList{B}{ii}{\alpha}` (with the package option `[greek]`).
 - β this is the item `myList{B}{ii}{\beta}` (or `innerlist{ii}{\beta}`).
- C) this is the item `myList{C}` (as you'd have expected...)
- perso** this is the item `myList{perso}`.
- D) this is the item `myList{D}` and is also `myList,5` in the *commaref* scheme.
- this is the item `myList{D}{1}`
 - this is the item `myList{D}{2}`
- inside *itemize* lists, labels are replaced by their arabic index in the *refitem* scheme.

And here is a reference to:

<code>\refitem{myList{B}{ii}{\alpha}}</code>	α
<code>\refitem{myList{B}{ii}{\alpha}}[\label\ \label\ \label]</code>	B) ii) α
<code>\refitem{myList{B}{ii}{\alpha}}[\label*---\label*---\label]</code>	B–ii– α

Now what happens with a description list:

- 1°) this is the item `myList{1}`
- 2°) this is the item `myList{2}`
- First description label** this is the item `myList{2}{first description label}`
- And a second one** this is the item `myList{2}{and a second one}`
- 3°) this is the item `myList{3}`

<code>\refitem{myList{2}{first description label}}</code>	First description label
<code>\refitem{myList2{2}{first description label}}[\ref*, \label]</code>	2, First description label
<code>\refitem{myList2{2}{first description label}}[\ref\ \ref]</code>	2°) First description label

The correspondance between the reference name and the reference that is finally printed is immediate: the *refitem* scheme is almost useful when used with *hyperref*.

The list name `myList` is case sensitive, but the item number `{B}` or `{first description label}` is not. This can be changed with the *CaSeS* package option. Trailing spaces in the item label are always removed from the reference name in the *refitem* scheme.

1.2.3 The *user* scheme: with the `\label` command

- 1 – this is the item `myList,1` and `myList{1}` and as long as I type `\label{Important Item}` it is also named *Important Item*.
- 2 – this is the item `myList,2` and `myList{2}`.
- a) this is the item `myList,2,1` and `myList{2}{a}`.
 - b) this is the item `myList,2,2` and `myList{2}{b}` and as long as I type: `\label{Important subitem}` it is also named *Important subitem*.

Then you get:

<code>\refitem{Important subitem}</code>	b)
<code>\refitem{Important subitem}[\label\ \label]</code>	2 – b)
<code>\refitem{Important subitem}[\ref* sub point \ref]</code>	2 sub point b)

The reference is case sensitive.

2 User Interface

2.1 Naming lists

To use the reference schemes defined by `enumitem-zref`, it is necessary to give lists a name. For that purpose, two new keys: `name` and `name*` have been added to the keys provided by `enumitem`.

The key ‘`name`’ gives a name to the list, while `name*` is a bit more clever and should be used in every circumstances when lists are nested, almost when lists are automatically named by the mean of the paragraph they belong to. Example:

```
\newlist{myList}{enumerate}{3}
\setlist{myList}{label* = \arabic*., name* = \arabic{section}.\arabic{subsection}}
```

In case of nested lists, `name*` will compare the name you gave to the list with the name of its parent list, and ignore the key in case of equality.

This gives:

1. First item
2. Second item
 - 2.1. sub item
 - 2.2. another sub item
 - 2.3. a last sub item \label{for further reference}

And then:

<code>\refitem{2.1,2,3}</code>	2.3.	in the <i>commaref</i> scheme
<code>\refitem{2.1{2}{3}}</code>	2.3.	in the <i>refitem</i> scheme
<code>\refitem{for further reference}</code>	2.3.	in the <i>user</i> scheme

To name your lists automatically it is also possible to use (these are examples):

- `interfaces-hypbmsec` with package `hypbmsec`:
`name* = \csname the\lastsectionname\endcsname`
- package `refcount`: `name* = \getrefbykeydefault{reference}{anchor}{default}`.

2.2 Referencing the items : `\refitem`

`\refitem{⟨reference⟩}[text with properties \ref, \label etc..]`^{babel}
`\refitem*`

`enumitem-zref` provides `\refitem` to reference the items of lists with the following possible syntaxes:

<code>\refitem{refname}[properties]</code>	displays a reference to <code>refname</code> the text displayed depends on the optional <code>properties</code> whose default value is <code>\ref</code> .
.....	
<code>\refitem* {refname}[properties]</code>	can be used when you want a reference but no hyperlink.

If the reference `refname` does not point to an item, `\refitem` behaves like `\ref` (or `\ref*`) and the `[properties]` are ignored. A warning is displayed in the `.log` file.

The properties available with `\refitem` are:

- | | |
|--------------------------|---|
| <code>\ref</code> | This is the default property: the same as L ^A T _E X’s <code>\ref</code> command: it expands to the text corresponding to the <code>ref</code> key of the <code>enumitem</code> list environment, which by default is: <ul style="list-style-type: none"> • in <code>enumerate</code> lists: the same as the <code>label</code>. • in <code>itemize</code> lists: the index of the item. • in <code>description</code> lists: the user defined label, or if no label has been given to the <code>\item</code> command, the index of the item. |
|--------------------------|---|

<code>\label</code>	The label is printed, whatever value the <code>ref</code> key has been given. For <code>description</code> , the label may be empty.
<code>\label*</code> <code>\ref*</code>	This is a variant of <code>\label</code> : automatic labels in <code>enumerate</code> lists are related to a counter. <code>\ref*</code> or <code>\label*</code> prints the formatted value of the counter, without any text around. For <code>itemize</code> , <code>description</code> or user items inside <code>enumerate</code> lists, this is the same as <code>\label</code> .
<code>\page</code>	The same as <code>\pageref</code> .
<code>\name</code>	The name that has been given to the list.
<code>\depth</code>	The depth of the list (equal to 1 if the list is not nested).
<code>\index</code>	The index of the item; user defined items are counted.
<code>\listctr</code>	The item number inside numbered lists (<i>ie.</i> <code>enumerate</code>); user defined items are not counted: this is the value of the counter <code>\@listctr</code> .
<code>\type</code>	The type of the list (<i>ie.</i> <code>enumerate</code> , <code>itemize</code> or <code>description</code>)
<code>\refitem</code>	The list of references names that can be invoked with <code>\refitem</code> for this specific item in the <i>refitem</i> scheme (see also <code>\zeninfo</code>).
<code>\commaref</code>	The list of references names that can be invoked with <code>\refitem</code> for this specific item in the <i>commaref</i> scheme (see also <code>\zeninfo</code>).

`\refitem[file:...]{{reference}}{{text}}babel`

For reference to external files (to make hyperlinks), `\refitem` can be invoked with the following syntax:

<code>\refitem[file:...]{refname}{text}</code>	Can be used for references to an external .pdf file.
<code>\refitem[]{refname}{text}</code>	The current file is used by default...

Only the *refitem* and the *commaref* schemes can be used. The *user* scheme cannot be used presently for external files. An implementation “à la xr” might be provided in a further release.

2.3 enumitem-zref package options

enumitem-zref can be loaded with the following options:

↑ defaults ↓	true	<code>enumerate</code>	disables the references for <code>itemize</code> and <code>description</code> lists: only enumerated lists can be referenced.
	true	<code>itemize</code>	enables/disables the references for <code>itemize</code> lists.
	true	<code>description</code>	enables/disables the references for <code>description</code> lists.
	false	<code>greek</code>	Defines <code>\engrec</code> (lowercase) and <code>\EnGrec</code> (uppercase) to make <code>enumerate</code> lists of items with the greek alphabet as the label (packageengrec). <code>\engrec</code> and <code>\EnGrec</code> can be used in place of <code>\arabic</code> (f.ex. <code>label=\engrec*</code>).
	false	<code>CaSeS</code>	Activate this option to get CaSeS sEnSiTiV references in the <i>refitem</i> scheme. This option applies only to the <i>refitem</i> scheme. List names are always case sensitive. If this option is enabled with <code>hyperref</code> , anchors corresponding to the <i>refitem</i> scheme will be case sensitiv as well. Package <code>hyperref</code> ’s <code>[verbose]</code> option allows to see anchor names in the .log file.

	CaSeS = true	CaSeS = false
	I) First	I) First
	II) Second	II) Second
<code>\refitem{myList{I}}</code>	✓works	✓works
<code>\refitem{myList{i}}</code>	✗doesn’t work	✓works
<code>\refitem{mylist{I}}</code>	✗doesn’t work	✗doesn’t work
List names are always case sensitive.		

Package option	Types of lists that can be referenced		
	<i>enumerate</i>	<i>itemize</i>	<i>description</i>
[enumerate]	•		
[itemize]	•	•	
[description]	•		•
[enumerate,itemize]	•	•	
[enumerate,description]	•		•
[itemize,description]	•	•	•
[enumerate,itemize,description]	•	•	•
none of the above	•	•	•

2.4 If your are lost \zeninfo is your friend

\zeninfo[⟨info-key⟩]

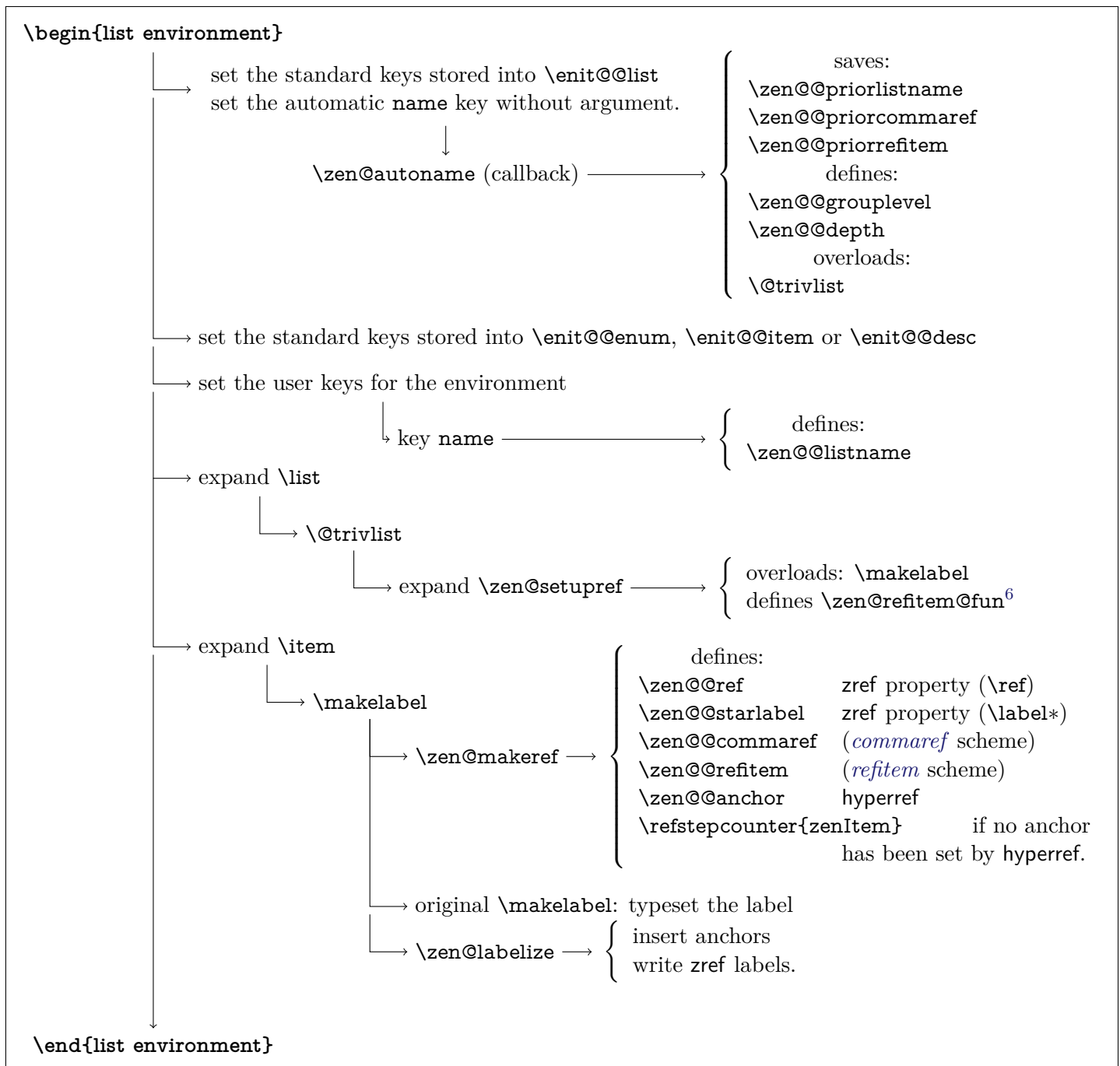
In the paragraph following an item, `\zeninfo` reports some informations about the references to the current item:

- depth** The depth of the list.
- index** The index of the item.
- commaref** The references that can be used to reference the item in the *commaref* scheme.
- refitem** The reference that can be used to reference the item in the *refitem* scheme.
- anchor** The anchor set for the item (if hyperref).

With no argument, `\zeninfo` prints all these informations.

5. For the French language with `babel frenchb-setup`, `itemize` lists can be processed by `enumitem-zref` only if the option:
`\frenchbsetup{CompactItemize=false}`
has been set. This option is automatically set by `enumitem-zref` when option `[itemize]` is passed.

3 Flow chart of expansion: How does it work ?



6. Used for the property `\label*` and to define the label (and the anchor) in the *refitem* scheme.

4 Implementation

4.1 Identification and requirements

- `enumitem-zref` is intended to use with \LaTeX so we don't check if it is loaded twice.
- The package namespace is `\zen@`

```

1 \*package\
2 \NeedsTeXFormat{LaTeX2e}[2005/12/01]
3 \ProvidesPackage{enumitem-zref}[2010/12/10 v1.2 - Extended references for enumitem pkg]
4 \RequirePackage{ltxcmds,etoolbox,kvoptions,getttitlestring,enumitem}
5 \RequirePackage{zref}[2010/05/01]
6 \RequirePackage{zref-user,zref-counter}
7 \GetTitleStringDisableCommands{\let\enit@format \@empty}

```

Some `\catcode` assertions:

```

8 \let\zen@AtEnd \@empty
9 \def\TMP@EnsureCode#1#2{\{%
10 \count@\catcode#1\relax
11 \toks@\expandafter{\zen@AtEnd \catcode#1 }
12 \xdef\zen@AtEnd{\the\toks@ \the\count@\relax}}
13 \catcode#1 #2\relax
14 }
15 \TMP@EnsureCode{40}{12} % (
16 \TMP@EnsureCode{41}{12} % )
17 \TMP@EnsureCode{'\I}{14} % I (default comment: itemize option)
18 \TMP@EnsureCode{'\D}{14} % D (default comment: description option)
19 \TMP@EnsureCode{'\E}{9} % E (default ignore: external option)

```

4.2 The package options

```

20 \SetupKeyvalOptions{family=zen,prefix=zen@}
21 \DeclareBoolOption{CaSeS}
22 \DeclareBoolOption{itemize}
23 \DeclareBoolOption{description}
24 \DeclareBoolOption{enumerate}
25 \DeclareVoidOption{greek}{\%
26 \AtEndOfPackage{\%
27 \RequirePackage{engrec,alphalph}
28 \let\zen@grecORI@grec \let\zen@GrecORI@Grec
29 \let\@grec\relax \let\@Grec\relax
30 \newalphalph\@grec[wrap]\zen@grecORI{24}
31 \newalphalph\@Grec[wrap]\zen@GrecORI{24}
32 \AddEnumerateCounter\engrec\@grec{\@grec{23}}\% widest is \psi
33 \AddEnumerateCounter\EnGrec\@Grec{\@Grec{12}}\% widest is M
34 }%
35 }
36 \DeclareVoidOption{external}{\catcode'\E 14\relax}\% E = comment
37 \DeclareDefaultOption{\@unknownoptionerror}
38 \ProcessLocalKeyvalOptions{zen}
39 \ifzen@enumerate\else
40 \ifzen@itemize \zen@enumeratetrue\else
41 \ifzen@description \zen@enumeratetrue\else
42 \zen@enumeratetrue \zen@itemizetrue \zen@descriptiontrue
43 \fi\fi
44 \ifzen@enumerate\else
45 \zen@enumeratetrue \PackageWarning{enumitem-zref}
46 {Option 'enumerate = false' has no effect}%
47 \fi
48 \fi
49 \ifzen@itemize \catcode'\I 9 \fi % ignore I
50 \ifzen@description\catcode'\D 9 \fi % ignore D
51 \AtBeginDocument{\zen@item@comma \setcounter{zenItem}\zen@}

```

Babel French: `frenchb` redefines `\itemize` at begin document.

Therefore, when option `[itemize]` of `enumitem-zref` is used, one have to ensure that babel-french setup `CompactItemize=false` is set:

```
52 \AtEndPreamble{\ifdefined\frenchbsetup\frenchbsetup{CompactItemize=false}\fi}% [itemize] option
```

4.3 Some constants

`zenItem` is the global L^AT_EX counter for zref labels and anchors added by `enumitem-zref`

`\c@zenidx` is the local counter for the index of the items. Every item is counted.

`\zen@toks` A token to be used by `\refitem` at the time the (optional) properties are analyzed: we can't use `\toks@` at this time, because `\toks@` could have been set to a value anywhere else.

```
53 \newcounter{zenItem}
54 \globcount\c@zenidx
55 \newtoks\zen@toks
```

`\zen@@autolabel` To check if the item user-defined or automatic.

```
56 \def\zen@@autolabel{\@itemlabel}
```

`\zen@@noreference` How to display undefined references.

```
57 \def\zen@@noreference{\nfss@text{\reset@font\bfseries ??}}
```

`\zen@Hy@anchor` is used to put the anchors for the items. It's a wrapper for `hyperref` `\Hy@raisedlink`.

```
58 \protected\def\zen@Hy@anchor#1{%
59   \Hy@raisedlink{\hyper@anchorstart{#1}\hyper@anchorend}%
60 }% \zen@Hy@anchor (wrapper for \Hy@raisedlink)
```

`\zen@lowercase` is used to make lowercase references in the `refitem` scheme, when the package option `CaSeS` is not `\zen@CaSeS` active.

```
61 \let\zen@CaSeS \@firstofone
62 \ifzen@CaSeS \let\zen@lowercase \zen@CaSeS
63 \else \let\zen@lowercase \lowercase
64 \fi
```

`\zen@ifrefundefined` Switch to the first part if the L^AT_EX label exists, to the second part otherwise.

```
65 \def\zen@ifrefundefined#1{\ltx@ifundefined{r@#1}}
```

`\zen@box` A utility to expand code inside a temporary `\vbox`

```
66 \def\zen@box{\hfuzz\maxdimen \vfuzz\hfuzz \hbadness\@M \vbadness\hbadness
67   \setbox\z@=\vbox}
```

4.4 Properties for the zref list: `zen@list`

`\zen@unique` is the unique counter to create unique labels for zref references (internal use only):

```
68 \zref@newlist{zen@list}
69 \def\zen@{zen>\the\c@zenItem}
70 \zref@newprop{zen@}[]\zen@
71 \def\zen@temp#1#2#3{%
72   \zref@newprop{zen@#2}[\zen@@noreference]{#3}%
73   \ifx U#1\expandafter\let\csname zen@extract@#2\endcsname\unexpanded
74   \else \expandafter\let\csname zen@extract@#2\endcsname\detokenize
75   \fi
76 }%
77 \zref@newprop*{zen@page}[\zen@@noreference]\thepage
78 \expandafter\let\csname zen@extract@page\endcsname\unexpanded
79 \zen@temp U{anchor}\zen@@anchor
80 \zen@temp U{default}\zen@@ref
81 \zen@temp U{type}{\ifdefined\zen@@type\ifnum\zen@@type<\z@\else \enit@type \fi\fi}
82 \zen@temp D{listname}\zen@@listname
83 \zen@temp U{depth}{\the\@listdepth}
```

```

84 \zen@temp U{index}{\the\c@zenidx}
85 \zen@temp U{listctr}{\ifdefined\@listctr \the\value\@listctr\fi}
86 \zen@temp U{parents}\zen@@parents
87 \zen@temp U{label}\zen@@itemlabel
88 \zen@temp U{starlabel}\zen@@starlabel
89 \zen@temp U{ref}\zen@@ref
90 \zen@temp U{item}\@itemlabel
91 \zen@temp U{currentlabel}\@currentlabel
92 \zen@temp U{currentlabelname}\@currentlabelname
93 \zen@temp D{refitem}\zen@@refitem
94 \zen@temp D{commaref}\zen@@commaref
95 \zen@temp U{counter}{\zref@getcurrent{counter}}
96 \zref@addprop{zen@list}{%
97   zen@default,zen@page,zen@type,zen@depth,zen@listctr,%
98   zen@listname,zen@index,zen@label,zen@ref,zen@starlabel,%
99   zen@parents,zen@refitem,zen@commaref,zen@item,zen@currentlabel,zen@counter}
100 \newif\ifzen@Hy
101 \AtBeginDocument{%
102   \ifdefined\hyper@anchor \zen@Hytrue
103     \zref@addprop{zen@list}{zen@anchor,zen@currentlabelname}%
104   \else \zen@Hyfalse \let\zen@refHy \zen@refText
105   \fi
106 }

```

4.5 Normalisation of references and item labels

\zen@Normalize

```

107 \def\zen@Normalize#1#2#3{\begingroup \let\zen@=#1%
108   \def\GetTitleStringResult{#3}\toks@{}}%
109   \ifx \GetTitleStringResult\@empty \else
110     \zen@NormalizeCommands \GetTitleStringExpand{#3}%
111     \expandafter\zen@trailingspaces \GetTitleStringResult{\@nnil}\@nnil
112   \fi
113   \edef\x{\endgroup
114     \def\noexpand#2{\detokenize\expandafter{\GetTitleStringResult}}%
115     \def\noexpand\zen@temp {\detokenize\expandafter{\the\toks@}}%
116   }\x
117 }% \zen@Normalize
118 \def\zen@trailingspaces#1#{\toks@{#1}\expandafter\zen@trailingsp@ces}
119 {\catcode47 = 8 % /
120 \gdef\zen@trailingsp@ces#1{%
121   \ifx \@nnil#1\@empty \expandafter\remove@to@nnil
122   \else \zen@{\toks@\expandafter{\the\expandafter\toks@
123     \expandafter{\romannumeral0\zen@postspace #1/ /}}}%
124     \expandafter\zen@trailingsp@ces % (loop)
125   \fi
126 }% \zen@trailingsp@ces
127 \gdef\zen@postspace#1 /{\zen@postsp@ce#1/}
128 \gdef\zen@postsp@ce#1/#2{ #1}
129 }% \catcode group
130 \def\zen@NormalizeCommands{%
131   \let\ensuremath \@firstofone % \ensuremath (engrec package)
132   \let\relax \@empty
133   \csname @safe@activetrue\endcsname
134 }% \zen@NormalizeCommands

```

4.6 Keys `name` and `name*` are added to `enumitem` environments

```

135 \define@key{enumitem}{name}[]{\zen@namelist{#1}}
136 \define@key{enumitem}{name*}[]{\zen@starnamelist{#1}}

```

`\zen@namelist` is the callback for the key `name`.

```

137 \E \AtBeginDocument{\gappto\enit@@list{,name}}% ignore if [external]
138 \def\zen@namelist #1{%
139   \ifdefined\zen@@grouplevel\else \let\zen@@grouplevel \z@ \fi
140   \ifnum\zen@@grouplevel<\currentgrouplevel% first (automatic) assignment to name
141     \zen@type % = (automatic from \enit@@list)
142     \ifcase\zen@type % (case 0) = enum
143       \zen@autoname
144     \or % (case 1) = itemize
145     \zen@autoname % [itemize] option only
146     \or % (case 2) = description
147     \zen@autoname % [description] option only
148   \fi
149   \else \zen@Normalize\zen@CaSeS\zen@@listname {#1}%
150     \let\zen@@listname \zen@temp
151   \fi
152 }% \zen@namelist (code for key name)

```

`\zen@starnamelist` is the callback for the key `name*`.

```

153 \def\zen@starnamelist #1{% (1st automatic assignment to name done before)
154   \zen@Normalize\zen@CaSeS\zen@@listname{#1}%
155   \let\zen@@listname \zen@temp
156   \ifx \zen@@priorlistname\zen@@listname
157     \let\zen@@listname \@empty \fi
158   \zen@namelist\zen@@listname
159 }% \zen@starnamelist (code for key name*)

```

`\zen@autoname` is the callback for the key `name` automatically set without argument by `\setkeys{enit@@list}`.

This corresponds to some setup:

- save the prior list name *etc.*.
- define the group level,
- define the list depth,
- reset the index,
- save the definition of `\@trivlist` and overload it temporarily until its next expansion at the beginning of the list.

```

160 \def\zen@autoname{% (1st automatic assignment to key name (by \enit@@list))
161   \unless \ifx\zen@@listname \@empty
162     \let\zen@@priorlistname \zen@@listname \fi
163   \unless \ifdefined\zen@@priorlistname \let\zen@@priorlistname \@empty \fi
164   \let\zen@@listname \@empty
165   \ifdefined\zen@@commaref \let\zen@@priorcommaref \zen@@commaref
166   \else \let\zen@@priorcommaref \@empty
167   \let\zen@@commaref \@empty
168   \fi
169   \ifdefined\zen@@refitem \let\zen@@priorrefitem \zen@@refitem
170   \else \let\zen@@priorrefitem \@empty
171   \let\zen@@refitem \@empty
172   \fi
173   \edef\zen@@grouplevel {\the\currentgrouplevel}%
174   \edef\zen@@depth {\number\@listdepth}%
175   \ifnum \zen@@depth>\@ne \edef\zen@@parents{\the\c@zenItem}\zen@@parents}%
176   \else \let\zen@@parents \@empty \fi
177   \let\zen@refitem@fun \@undefined
178   \c@zenidx \z@
179   \let\zen@trivlistORI \@trivlist
180   \let\@trivlist \zen@trivlist

```

```
181 }% \zen@name@list
```

`\zen@trivlist` `\@trivlist` is temporarily overloaded, and the original definition is immediately restored at the time it will be expanded, just after having set all the specified keys for the list environment.

The overload sets `\zen@setuprefs` to be expanded at the end of `\@trivlist`.

```
182 \def\zen@trivlist{%
183   \let\@trivlist \zen@trivlistORI    % restore original immediately
184   \@trivlist \zen@setuprefs
185 }% \zen@trivlist
```

`\zen@type` expands to:

- 0 if `enumerate`
- 1 if `itemize` (and `[itemize]` option is active)
- 2 if `description` (and `[description]` option is active)
- 1 in any other case

```
186 \def\zen@type{%
187   \ifdefined\enit@type
188     \expandafter\strip@prefix \if e\enit@type>\let\zen@@type\z@ % enum begins with e
189     \else>\expandafter\strip@prefix \if i\enit@type>\let\zen@@type\@ne % item begins with i
190     \else>\expandafter\strip@prefix \if d\enit@type>\let\zen@@type\tw@ % desc begins with d
191     \else>\let\zen@@type\m@ne
192     \fi\fi\fi
193   \else \let\zen@@type\m@ne
194   \fi
195 }% \zen@type
```

4.7 Overloading `\makelabel`

`\zen@setuprefs` expands just at the end of `\@trivlist`. `\makelabel` is overloaded in order to make the references and write the labels into the `.aux` file.

`\zen@makelabel`

```
196 \def\zen@setuprefs{%
197   \edef\zen@temp{\zen@@priorlistname\zen@@listname}%
198   \ifx \zen@temp\@empty
199     \ifx \makelabel\zen@makelabel \let\makelabel \zen@makelabelORI \fi
200   \else
201     \ifx \makelabel\zen@makelabel \else
202       \let \zen@makelabelORI \makelabel
203       \let \makelabel \zen@makelabel \fi
204   \fi
205   \let\zeninfo \zen@info
206   \zen@refitem@implicit
207 }% \zen@setuprefs
208 \def\zen@makelabel#1{\zen@makeref{#1}\zen@makelabelORI{#1}\zen@labelize}
```

`\zen@refitem@implicit` defines `\zen@refitem@fun` which cancels text around the formatted counter in `enumerate` lists (*ie.* numbered lists). This will be used at the time of `\makelabel` to create the `refitem` and the property `\label*` (which corresponds to `zref` property `zen@starlabel`).

```
209 \def\zen@refitem@implicit{% determine value of key refitem according to the label
210   \def\zen@gtemp {\the\c@zenidx}%
211   \ifnum \zen@@type=\z@ % enumerate
212     {\let\enit@refstar \zen@refitemfromlabel
213      \enit@labellist \zen@box{\@itemlabel}}}%
214   \fi
215   \let\zen@refitem@fun \zen@gtemp
216 }% \zen@refitem@implicit
217 \def\zen@refitemfromlabel#1{\def#1##1{%
218   \def\zen@temp{##1}\ifx \zen@temp\@enumctr
219     \gdef\zen@gtemp{{#1{##1}}}\fi}%
220 }% \zen@refitemfromlabel
```

4.8 Creating references

`\zen@makeref` is expanded just before the original version of `\makelabel`

The aim is to set the `zref` properties defined by `enumitem-zref` for the current item, and to define the anchor names.

```

221 \def\zen@makeref#1{% #1 = argument of \makelabel
222   \def\zen@@itemlabel{#1}\advance\c@zenidx \@ne
223   \ifx \zen@@itemlabel\zen@@autolabel
224     \if@nmbrrlist
225       \global\advance\c@zenItem \@ne
226       \let\zen@@ref \@currentlabel
227     \else
228       {\refstepcounter{zenItem}}%
229       \let\zen@@ref \zen@refitem@fun
230     \fi
231     \let\zen@@starlabel \zen@refitem@fun
232     \zen@Normalize\zen@lowercase\zen@currentrefitem \zen@refitem@fun
233   \else % user defined label
234     {\refstepcounter{zenItem}}%
235     \zen@Normalize\zen@lowercase\zen@@ref {\zen@@itemlabel}%
236     \let\zen@@starlabel \zen@@ref
237   \fi
238   \let\zen@currentrefitem \zen@temp % \zen@temp defined by \zen@Normalize
239   \let\zen@@anchor \@currentHref
240   \zen@makeref@item@comma
241 }% \zen@makeref

```

`\zen@labelize` is expanded just after the original version of `\makelabel`

`zlabels` are written into the `.aux` file.

```

242 \def\zen@labelize{%
243   \ifzen@Hy \zen@anchors
244     \zref@labelbyprops{zen->\@currentHref}{zen@}\fi
245   \zref@labelbylist\zen@{zen@list}%
246   \zen@keeplabel
247 }% \zen@makeref

```

`\zen@makeref@item@comma` Creates the references for the *commaref* and the *refitem* schemes.

```

248 \def\zen@makeref@item@comma{%
249   \begingroup % to avoid silent assignment to \relax
250   \edef\x{\endgroup
251     \def\noexpand\zen@@commaref{%
252       \expandafter\zen@commaref@\zen@@priorcommaref \@nnil}%
253     \def\noexpand\zen@@refitem{%
254       \expandafter\zen@refitem@\zen@@priorrefitem \@nnil}%
255   }\x
256 }% \zen@makeref@item@comma
257 \def\zen@refitem@#1{%
258   \ifx \@nnil#1%
259     \ifx \zen@@listname\@empty \else
260       {\zen@@listname\zen@currentrefitem}\fi
261   \else {#1\zen@currentrefitem}\expandafter\zen@refitem@ % loop
262   \fi
263 }% \zen@refitem@
264 \def\zen@commaref@#1{%
265   \ifx \@nnil#1%
266     \ifx \zen@@listname\@empty \else
267       {\zen@@listname,\the\c@zenidx}\fi
268   \else {#1,\the\c@zenidx}\expandafter\zen@commaref@ % loop
269   \fi
270 }% \zen@commaref@

```

`\zen@anchors` Put the PDFanchors for the current item.

```

271 \let\zen@list@anchors\@empty
272 \def\zen@anchors{\begingroup
273   \def\zen@@anchors{\endgroup}%
274   \expandafter\expandafter\expandafter\zen@anchors@
275     \expandafter\zen@@commaref \zen@@refitem \@nnil
276   \zen@@anchors
277 }% \zen@anchors
278 \def\zen@anchors@#1{%
279   \ifx \@nnil#1\else
280     \expandafter\in@\csname refitem.#1\expandafter\endcsname\expandafter{\zen@list@anchors}%
281     \ifin@ \zen@anchor@ignore{#1}%
282     \else
283       \edef\zen@@anchors{\zen@@anchors\zen@Hy@anchor{refitem.#1}}%
284       \xdef\zen@list@anchors{%
285         \unexpanded\expandafter{\zen@list@anchors}%
286         \expandafter\noexpand\csname refitem.#1\endcsname}%
287       \fi
288       \expandafter\zen@anchors@ % loop
289     \fi
290 }% \zen@anchors@ (loop)
291 \def\zen@anchor@ignore#1{\PackageInfo{enumitem-zref}{Ignored anchor '#1'}}

```

`\zen@keeplabel` As far as references are built and written inside `\makelabel` (which was overloaded) they are built inside a group (inside a `\sbox`, and thus inside a `\color@setgroup... \color@endgroup` group if packages `color` or `xcolor` are used, and may be inside other grouping levels depending on the user need (she may have redefined `\makelabel` for a special purpose).

Therefore, we take advantage of the e-TeX command `\currentgrouplevel` in order to decide how many levels of grouping we have to “pass through” for keeping the current values of `\z@zenidx` (the item *index*), `\zen@@refitem`, and `\zen@@commaref`.

The “target” group level has been saved into `\zen@@grouplevel` at the beginning of the environment, while setting the automatic `name` key for the first time.

```

292 \def\zen@keep#1{\def\noexpand#1{\unexpanded\expandafter{#1}}}%
293 \def\zen@keeplabel{%
294   \xdef\zen@gtemp{%
295     \noexpand\ifnum \zen@@grouplevel<\currentgrouplevel
296       \aftergroup \noexpand\zen@gtemp
297     \noexpand\else
298       \c@zenidx \the\c@zenidx\relax
299       \zen@keep \zen@@refitem
300       \zen@keep \zen@@commaref
301       \ifzen@Hy \zen@keep\zen@@anchor\fi
302     \noexpand\fi
303   }\aftergroup\zen@gtemp
304 }% \zen@keeplabel

```

4.9 Extracting zref properties

`\zen@extract` A wrapper to `\zref@extractdefault` which can also `\detokenize` the result.

```

305 \newcommand*\zen@extract[3]{% #1=ref #2=prop #3=def
306   \csname zen@extract@#2\endcsname\expandafter\expandafter%
307     \expandafter{\zref@extractdefault{zen>\number#1}{zen@#2}{#3}}%
308 }% \zen@extract

```

4.10 Referencing the items: \refitem

`\refitem` The general macro to print references and insert hyperlinks, with a star variant (no hyperlink).

```

309 \newrobustcmd*\refitem{\begingroup \ifstar
310   {\zen@refitemOpt \zen@reftext }% no hyperlink
311   {\zen@refitemOpt \zen@refHy }%
312 }% \refitem
313 \def\zen@refHy{%

```

```

314 %% \edef\zen@@anchor{\zen@extract\c@zenItem{anchor}}{\unexpanded}%
315 %% \ifx \zen@@anchor\@empty \zen@@text
316 %% \else
317 \hyperref{}{}{\zref@extract{zen>\the\c@zenItem}{zen@anchor}}\zen@@text
318 %% \fi
319 }% \zen@ref
320 \def\zen@refText{\zen@@text}

```

`\zen@refitemOpt` If the first argument is an option, then `\refitem` makes a hyperlink to an external document.

```

321 \def\zen@refitemOpt#1{\let\zen@ref@#1\@ifnextchar[\zen@refitemExternal \zen@refitemRef }
322 \def\zen@refitemRef #1{\@tempswattrue \let\zen@lastwarn \m@ne
323 \zen@Normalize\zen@lowercase\zen@@label {#1}%
324 \zen@ifrefundefined \zen@@label
325 {\expandafter\expandafter\expandafter \zen@refused
326 \csname\ifcsname zen@refitem(\zen@temp)\endcsname
327 zen@refitem(\zen@temp)\else
328 zen@norefitem\fi\endcsname>}%
329 \zen@refitemFromAnchor
330 }% \zen@refitemRef
331 \def\zen@norefitem {\z@>0}

```

`\zen@refused` Checks if the label exists in the *commaref* scheme or in the *refitem* scheme.

```

332 \def\zen@refused#1>#2>{%
333 \ifcase#1\relax
334 \@tempswafalse \refused\zen@@label
335 \let\zen@ref@ \zen@refText % do not create a hyperlink if no reference
336 \or \let\zen@@label \zen@temp \c@zenItem ==#2\relax
337 \else \let\zen@@label \zen@temp \zen@MultipleLabels
338 \c@zenItem ==#2\relax
339 \fi
340 \zen@refitem@prop
341 }% \zen@refused

```

`\zen@refitemFromAnchor` When `\refitem` refers to a user item, we can reached the `zref` properties with the chain:

LaTeX label \longrightarrow hyperref anchor \longrightarrow enumitem-zref label \longrightarrow property

```

342 \def\zen@refitemFromAnchor{%
343 \edef\zen@@anchor{\getrefbykeydefault\zen@@label{anchor}}{\}}%
344 \ifx \zen@@anchor\@empty
345 \@tempswafalse \G@refundefinedtrue \zen@warn\z@
346 \let\zen@ref@ \zen@refText
347 \else
348 \edef\zen@temp{\zref@extractdefault{zen->\zen@@anchor}{zen@}{}}%
349 \ifx \zen@temp\@empty
350 \@tempswafalse \G@refundefinedtrue \zen@warn\@ne
351 \ifzen@Hy \ifx\zen@ref@ \zen@refHy \def\zen@ref@{\ref{\zen@@label}}%
352 \else \def\zen@ref@ {\ref*{\zen@@label}}%
353 \fi
354 \else \def\zen@ref@{\ref{\zen@@label}}%
355 \fi
356 \else \c@zenItem \expandafter\strip@prefix\zen@temp\relax
357 \fi
358 \fi
359 \zen@refitem@prop
360 }% \zen@refitemFromAnchor

361 \def\zen@warn#1{\let\zen@lastwarn#1%
362 \message{Package enumitem-zref Warning:
363 \ifcase#1 %
364 \string\refitem{\zen@@label}:
365 User references to \string\label\space^^J\zen@spaces
366 require hyperref and the appropriate options for enumitem-zref^^J\zen@spaces
367 \ifzen@Hy Check the options given to package enumitem-zref.
368 \else Package hyperref is not loaded.\fi

```

```

369 \or User reference does not point to an item!^^J\zen@spaces
370 \string\refitem{\zen@@label}
371 \fi\on@line^^J^^J}}
372 \edef\zen@spaces{\@spaces\@spaces}

```

`\zen@refitem@prop` tests if `\refitem` has a following optional argument: the properties.

As long as the optional argument is in last position, we have to take a special care about the the spaces (`\@ifnextchar` removes spaces).

```

373 \def\zen@refitem@prop{\futurelet\zen@temp \zen@refitemProp}
374 \def\zen@gobblespace#1 {#1}%
375 \let\zen@space \relax
376 \def\zen@refitemProp{%
377 \ifcase 0\ifx \zen@temp[\else
378 \ifx \zen@temp\@sptoken 1\else
379 2\fi\fi\relax
380 \expandafter \zen@refitemPrint
381 \or \let\zen@space\space
382 \expandafter\zen@gobblespace\expandafter\zen@refitem@prop
383 \else \expandafter\zen@refitemPrint \expandafter[\expandafter]%
384 \fi
385 }% \zen@refitemProp

```

`\zen@refitemPrint` is the final macro to print the reference made by `\refitem`. The properties, if specified, are first analyzed by `\zen@refitemText`.

```

386 \long\def\zen@refitemPrint[#1]{\def\zen@@text{#1}%
387 \if@tempswa % got a reference
388 \ifx \zen@@text\@empty
389 \edef\zen@@text{\zen@extract\c@zenItem{default}\zen@@noreference}\zen@space}%
390 \else\zen@refitemText
391 \fi
392 \else
393 \ifx \zen@@text\@empty \def\zen@@text{\zen@@noreference\zen@space}%\refused\zen@@label
394 \else \zen@refitem@noText
395 \fi
396 \fi
397 \zen@ref@
398 \endgroup
399 }% \zen@refitemPrint

```

`\zen@refitemText` creates the text to be typeset for the reference, depending on the properties given by the user as the last optional argument to `\refitem`.

```

400 \def\zen@refitemText{\zen@propcount
401 \ifnum \c@zenidx>\z@ \advance\c@zenidx \m@ne
402 \ifnum \c@zenidx>\z@
403 \edef\zen@@parents{\zen@extract\c@zenItem{parents}}}%
404 \edef\zen@@depth{\zen@extract\c@zenItem{depth}0}%
405 \else \let\zen@@depth \@ne % shortcut (only one property)
406 \fi
407 \def\ref{\@ifstar{\zen@ancestorprop{starlabel}}{\zen@ancestorprop{ref}}}%
408 \def\label{\@ifstar{\zen@ancestorprop{starlabel}}{\zen@ancestorprop{label}}}%
409 \def\default{\zen@ancestorprop{default}}\def\counter{\zen@ancestorprop{counter}}%
410 \def\type{\zen@ancestorprop{type}}\def\listname{\zen@ancestorprop{listname}}%
411 \def\index{\zen@ancestorprop{index}}\def\listctr{\zen@ancestorprop{listctr}}%
412 \def\depth{\zen@ancestorprop{depth}}\def\refitem{\zen@ancestorprop{refitem}}%
413 \def\anchor{\zen@ancestorprop{anchor}}\def\commaref{\zen@ancestorprop{commaref}}%
414 \def\page{\zen@ancestorprop{page}}\def\currentlabel{\zen@ancestorprop{currentlabel}}%
415 \def\currentlabelname{\zen@ancestorprop{currentlabelname}}%
416 {\zen@box{\zen@@text
417 \expandafter}\expandafter}\expandafter\zen@toks
418 \expandafter{\the\zen@toks \zen@nil}%
419 \def\ref{\the\zen@toks}\zen@clprops
420 \def\ref{\@ifstar{\the\zen@toks}{\the\zen@toks}}\let\label\ref
421 \long\def\zen@##1##2\zen@nil{\zen@toks{##2\zen@nil}##1}\let\zen@nil\relax

```

```

422 \fi
423 }% \zen@refitemText
424 \def\zen@ancestorprop#1{%
425 \ifnum \zen@@depth>\c@zenidx
426 \advance\c@zenidx \m@ne
427 \ifnum \c@zenidx<\z@ \let\zen@ \c@zenItem
428 \else \edef\zen@{\zen@ancestor}%
429 \fi
430 \edef\zen@temp{{\zen@extract\zen@{#1}\zen@@noreference}}%
431 \else \def\zen@temp{{}}\advance\c@zenidx \m@ne
432 \fi
433 \zen@toks\expandafter{\the\expandafter\zen@toks \expandafter\zen@ \zen@temp}%
434 }% \zen@ancestorproperty
435 \def\zen@ancestor{\expandafter\ltx@car\romannumeral0\expandafter\ltx@GobbleNum
436 \expandafter{\the\expandafter\c@zenidx\expandafter}\zen@@parents{}{}{} \@nil % no need of {}...
437 }% \zen@getancestor
438 \def\zen@propcount{\c@zenidx\z@
439 \def\ref{\advance\c@zenidx \@ne}\zen@clprops
440 {\zen@box{\zen@@text \expandafter}\expandafter}\expandafter
441 \c@zenidx \the\c@zenidx\relax
442 }% \zen@propcount
443 \def\zen@clprops{\let\label\ref \let\page\ref
444 \let\none\ref \let\default\ref \let\counter\ref \let\listname\ref
445 \let\anchor\ref \let\depth\ref \let\type\ref \let\index\ref \let\listctr\ref
446 \let\refitem\ref \let\commaref\ref \let\currentlabel\ref \let\currentlabelname\ref
447 }% \zen@clprops
448 \def\zen@refitem@noText{\let\ref \zen@@noreference \zen@clprops}

```

`\zen@refitemExternal` for references to an anchor in an external document with `\hyperref`:

```

449 \def\zen@refitemExternal[#1]#2{%
450 \toks@{#1}\zen@Normalize\zen@lowercase\zen@@label {#2}%
451 \edef\zen@ref@{\endgroup
452 \noexpand\hyperref{\the\toks@}{refitem}{\zen@@label}%
453 }\zen@ref@
454 }% \zen@Hy@external

```

4.11 Undefined references and duplicate labels management

`\zen@MultipleLabels`

```

455 \def\zen@MultipleLabels{%
456 \PackageWarning{enumitem-zref}
457 {'\zen@@label' on page \thepage \space multiple defined\MessageBreak
458 Only the first one will be referenced}% et mettre ici la liste des références correspondant aux alias
459 }% \zen@MultipleLabels

```

4.12 `\zeninfo` (If you are lost...)

`\zen@info` `\zeninfo` executes `\zen@info` when inside environments:

```

460 \newrobustcmd*\zen@info[1][]{\begingroup \sloppy
461 \def\sep##1{##1\hskip\z@ plus.001fil\relax\linebreak[2]}%
462 \ifcase\zen@whichinfo{#1} \zen@@depth % case 0 (depth)
463 \or \the\c@zenidx % case 1 (index)
464 \or % case 2 (type)
465 \ifcase\zen@@type enumerate\or itemize\or description\or unknown\fi
466 \or anchor=\ifzen@Hy\zen@@anchor\else no hyperref\fi % case 3 (anchor)
467 \or refitem=\sep{}\detokenize\expandafter{\zen@@refitem}% case 4 (refitem)
468 \or commaref=\sep{}\detokenize\expandafter{\zen@@commaref}% case 5 (commaref)
469 \else % (case -1) give all infos
470 {\ifdefined\scriptsize\scriptsize\fi
471 zeninfo:\sep{}depth=\sep{}\zen@@depth
472 \sep/ idx=\sep{}\the\c@zenidx
473 \ifcase\zen@@type \sep/ ctr=\sep{}\the\value\@listctr \fi % enumerate only
474 \sep/ commaref=\sep{}\detokenize\expandafter{\zen@@commaref}%

```

```

475         \sep/ refitem=\sep{}\detokenize\expandafter{\zen@@refitem}%
476     \ifzen@Hy\sep/ anchor=\sep{}\zen@@anchor \fi
477 }%
478 \fi\endgroup
479 }% \zen@info

```

`\zeninfo` is executed when out of scope:

```

480 \newrobustcmd\zeninfo[1][\begingroup
481   \ifdefined\zen@@type
482   \ifcase \zen@@type \zeninfo@err
483   I \or \zeninfo@err
484   D \or \zeninfo@err
485   \else\zeninfo@err[{in environment \@currenvir\MessageBreak
486     You should be in a list environment}]%
487   \fi\endgroup
488 }% \zeninfo (out of scope)
489 \newcommand*\zeninfo@err[1][\{you should give a name to your \@currenvir\space list\}]{%
490   \PackageError{enumitem-zref}
491     {\string\zeninfo\space is not available\MessageBreak
492     #1}\@eha
493 }% \zeninfo@err

```

`\zen@whichinfo`

```

494 \def\zen@whichinfo#1{%
495   \expandafter\strip@prefix \if d#1>\z@ % depth (begins with d)
496   \else>\expandafter\strip@prefix\if i#1>\@ne % index (begins with i)
497   \else>\expandafter\strip@prefix\if t#1>\tw@ % type (begins with t)
498   \else>\expandafter\strip@prefix\if a#1>\thr@@ % anchor (begins with a)
499   \else>\expandafter\strip@prefix\if r#1>4 % refitem(begins with r)
500   \else>\expandafter\strip@prefix\if c#1>5 % commaref(begins with c)
501   \else>\m@ne
502   \fi\fi\fi\fi\fi
503 }% \zen@whichinfo

```

4.13 Create the references At Begin Document for the *refitem* and the *commaref* schemes

`\zen@item@comma`

```

504 \def\zen@item@comma{%
505   \begingroup \c@zenidx \@ne
506   \@whilesw \ifcsname Z@R@zen>\the\c@zenidx\endcsname\fi
507     {\advance\c@zenidx \@ne}%
508   \@whilenum \c@zenidx>\@ne
509   \do{\advance \c@zenidx \m@ne
510     \zref@wrapper@unexpanded{%
511       \edef\zen@refitem{%
512         \zref@extractdefault{zen>\the\c@zenidx}{\zen@refitem}{}}%
513         \zref@extractdefault{zen>\the\c@zenidx}{\zen@commaref}{}}}%
514     \edef\zen@refitem{\expandafter
515       \zen@item@comma@ \zen@refitem \@nnil}%
516     \expandafter\zen@def@item@comma@ \zen@refitem \@nnil}}%
517   \endgroup
518 }% \zen@item@comma
519 \def\zen@item@comma@ #1{%
520   \ifx\@nnil#1\else
521     \expandafter\noexpand\csname zen@refitem(\detokenize{#1})\endcsname
522     \expandafter\zen@item@comma@ % (loop)
523   \fi
524 }% \zen@item@comma@

```

`\zen@def@item@comma@`

```

525 \def\zen@def@item@comma@ #1{\ifx\@nnil#1\else
526   \zen@def@item@comma@{#1}\expandafter\zen@def@item@comma@ \fi % loop

```

```

527 }% \zen@def@item@comma@
\zen@def@item@comma

528 \newif\ifzen@duplicatealiases
529 \def\zen@def@item@comma #1{%
530   \xdef#1{%
531     \ifdefined#1%
532       \ifx\relax#1\@empty      \@ne>\the\c@zenidx% create new
533     \else \expandafter
534       \ifnum\expandafter\strip@prefix#1=\c@zenidx% redefinition to the same location
535         #1%
536       \else
537         \number\numexpr1+#1% increment multiplicity
538       \fi
539     \fi
540   \else      \@ne>\the\c@zenidx% create new
541   \fi}%
542 \ifzen@duplicatealiases\else
543 \expandafter\expandafter\expandafter\strip@prefix
544   \expandafter\strip@prefix\ifnum\@ne<#1>\global\zen@duplicatealiasestrue\else>>\fi
545 \fi
546 }% \zen@def@item@comma

547 \zen@AtEnd \let\zen@AtEnd\@undefined
548 \endpackage

```

References

- [1] *The enumitem package*; Javier Bezos
2009/05/18 v2.2 – Customized lists
CTAN:help/Catalogue/entries/enumitem.html
- [2] *The zref package*; Heiko Oberdiek
2010/05/01 v2.17 – New reference scheme for LaTeX2e
CTAN:help/Catalogue/entries/zref.html
- [3] *The gettitlestring package*; Heiko Oberdiek
2009/12/18 v1.3 – Cleanup title references
CTAN:help/Catalogue/entries/gettitlestring.html

History

[2010/12/10 v1.2]

- Bug encountered in `\zen@anchors@:`
`\ifx#1\@nnil` is not the same as `\ifx\@nnil#1` !!

[2010/12/02 v1.1]

- The first version.

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; plain numbers refer to the code lines where the entry is used.

Symbols	
<code>\@Grec</code>	28, 29, 31, 33
<code>\@currentHref</code>	239, 244
<code>\@currentlabel</code>	91, 226
<code>\@currentlabelname</code>	92
<code>\@currenvir</code>	485, 489
<code>\@enumctr</code>	218
<code>\@grec</code>	28, 29, 30, 32
<code>\@itemlabel</code>	56, 90, 213
<code>\@listdepth</code>	83, 174
<code>\@trivlist</code>	179, 180, 183, 184
<code>\@whilenum</code>	508
<code>\@whilesw</code>	506
<code>\currentgrouplevel</code>	140, 173, 295
<code>\currentlabel</code>	414, 446
<code>\currentlabelname</code>	415, 446
D	
<code>\DeclareBoolOption</code>	21, 22, 23, 24
<code>\DeclareDefaultOption</code>	37
<code>\DeclareVoidOption</code>	25, 36
<code>\default</code>	409, 444
<code>\define@key</code>	135, 136
<code>\depth</code>	412, 445
<code>\detokenize</code>	74, 114, 115, 467, 468, 474, 475, 521
E	
<code>\EnGrec</code>	33
<code>\engrec</code>	32
<code>\enit@list</code>	137, 141, 160
<code>\enit@format</code>	7
<code>\enit@labellist</code>	213
<code>\enit@refstar</code>	212
<code>\enit@type</code>	81, 187, 188, 189, 190
<code>\ensuremath</code>	131
F	
<code>\frenchbsetup</code>	52
<code>\futurelet</code>	373
G	
<code>\G@refundefinedtrue</code>	345, 350
<code>\getrefbykeydefault</code>	343
<code>\GetTitleStringDisableCommands</code>	7
A	
<code>\AddEnumerateCounter</code>	32, 33
<code>\aftergroup</code>	296, 303
<code>\anchor</code>	413, 445
<code>\AtEndOfPackage</code>	26
<code>\AtEndPreamble</code>	52
C	
<code>\c@zenidx</code>	53, 84, 178, 210, 222, 267, 268, 298, 401, 402, 425, 426, 427, 431, 436, 438, 439, 441, 463, 472, 505, 506, 507, 508, 509, 512, 513, 532, 534, 540
<code>\c@zenItem</code>	69, 175, 225, 314, 317, 336, 338, 356, 389, 403, 404, 427
<code>\commaref</code>	413, 446
<code>\count@</code>	10, 12
<code>\counter</code>	409, 444

<code>\GetTitleStringExpand</code>	110	S	
<code>\GetTitleStringResult</code>	108, 109, 111, 114	<code>\setcounter</code>	51
H		<code>\strip@prefix</code>	188, 189, 190, 356, 495, 496, 497, 498, 499, 500, 534, 543, 544
<code>\hbadness</code>	66	T	
<code>\hfuzz</code>	66	<code>\thepage</code>	77, 457
<code>\Hy@raisedlink</code>	59, 60	<code>\type</code>	410, 445
<code>\hyper@anchor</code>	102	V	
<code>\hyper@anchorend</code>	59	<code>\value</code>	85, 473
<code>\hyper@anchorstart</code>	59	<code>\vbadness</code>	66
<code>\hyperref</code>	317, 452	<code>\vbox</code>	67
I		<code>\vfuzz</code>	66
<code>\if@nmbrlist</code>	224	Z	
<code>\ifcsname</code>	326, 506	<code>\zen@</code> 69, 70, 107, 122, 245, 421, 427, 428, 430, 433	
<code>\ifin@</code>	281	<code>\zen@@anchor</code>	79, 239, 301, 314, 315, 343, 344, 348, 466, 476
<code>\ifzen@CaSeS</code>	62	<code>\zen@@anchors</code>	273, 276, 283
<code>\ifzen@description</code>	41, 50	<code>\zen@@autolabel</code>	56, 223
<code>\ifzen@duplicatealiases</code>	528, 542	<code>\zen@@commaref</code> 94, 165, 167, 251, 275, 300, 468, 474	
<code>\ifzen@enumerate</code>	39, 44	<code>\zen@@depth</code> 174, 175, 404, 405, 425, 462, 471	
<code>\ifzen@Hy</code> 100, 243, 301, 351, 367, 466, 476		<code>\zen@@grouplevel</code>	139, 140, 173, 295
<code>\ifzen@itemize</code>	40, 49	<code>\zen@@itemlabel</code>	87, 222, 223, 235
<code>\in@</code>	280	<code>\zen@@label</code>	323, 324, 334, 336, 337, 343, 351, 352, 354, 364, 370, 393, 450, 452, 457
<code>\index</code>	411, 445	<code>\zen@@listname</code> ... 82, 149, 150, 154, 155, 156, 157, 158, 161, 162, 164, 197, 259, 260, 266, 267	
L		<code>\zen@@noreference</code> .. 57, 72, 77, 389, 393, 430, 448	
<code>\label</code>	365, 408, 420, 443	<code>\zen@@parents</code>	86, 175, 176, 403, 436
<code>\listctr</code>	411, 445	<code>\zen@@priorcommaref</code>	165, 166, 252
<code>\listname</code>	410, 444	<code>\zen@@priorlistname</code>	156, 162, 163, 197
<code>\lowercase</code>	63	<code>\zen@@priorrefitem</code>	169, 170, 254
<code>\ltx@car</code>	435	<code>\zen@@cref</code>	80, 89, 226, 229, 235, 236
<code>\ltx@GobbleNum</code>	435	<code>\zen@@crefitem</code> 93, 169, 171, 253, 275, 299, 467, 475	
<code>\ltx@ifUndefined</code>	65	<code>\zen@@starlabel</code>	88, 231, 236
M		<code>\zen@@@text</code> 315, 317, 320, 386, 388, 389, 393, 416, 440	
<code>\makelabel</code>	199, 201, 202, 203, 221	<code>\zen@@@type</code>	81, 142, 188, 189, 190, 191, 193, 211, 465, 473, 481, 482
<code>\message</code>	362	<code>\zen@ancestor</code>	428, 435
N		<code>\zen@ancestorprop</code>	407, 408, 409, 410, 411, 412, 413, 414, 415, 424
<code>\newalphalph</code>	30, 31	<code>\zen@ancestorproperty</code>	434
<code>\newcounter</code>	53	<code>\zen@anchor@ignore</code>	281, 291
<code>\newtoks</code>	55	<code>\zen@anchors</code>	243, 271
<code>\nfss@text</code>	57	<code>\zen@anchors@</code>	274, 278, 288, 290
<code>\none</code>	444	<code>\zen@autoname</code>	143, 145, 147, 160
<code>\number</code>	174, 307, 537	<code>\zen@box</code>	66, 213, 416, 440
<code>\numexpr</code>	537	<code>\zen@CaSeS</code>	61, 149, 154
O		<code>\zen@clprops</code>	419, 439, 443, 447, 448
<code>\on@line</code>	371	<code>\zen@commaref@</code>	252, 264, 268, 270
P		<code>\zen@currentrefitem</code>	232, 238, 260, 261
<code>\PackageError</code>	490	<code>\zen@def@item@comma</code>	526, 528
<code>\PackageInfo</code>	291	<code>\zen@def@item@comma@</code>	516, 525
<code>\PackageWarning</code>	45, 456	<code>\zen@descriptiontrue</code>	42
<code>\page</code>	414, 443	<code>\zen@duplicatealiasestrue</code>	544
<code>\ProcessLocalKeyvalOptions</code>	38	<code>\zen@enumeratetrue</code>	40, 41, 42, 45
R		<code>\zen@extract</code>	305, 314, 389, 403, 404, 430
<code>\ref</code>	351, 352, 354, 407, 419, 420, 439, 443, 444, 445, 446, 448	<code>\zen@getancestor</code>	437
<code>\refitem</code>	5, 6, 309, 364, 370, 412, 446	<code>\zen@gobblespace</code>	374, 382
<code>\refstepcounter</code>	228, 234	<code>\zen@GrecORI</code>	28, 31
<code>\refused</code>	334, 393	<code>\zen@grecORI</code>	28, 30
<code>\remove@to@nnil</code>	121	<code>\zen@gtemp</code>	210, 215, 219, 294, 296, 303
<code>\romannumeral</code>	123, 435	<code>\zen@Hy@anchor</code>	58, 283

<code>\zen@Hy@external</code>	454	<code>\zen@refitemRef</code>	321, 322, 330
<code>\zen@Hyfalse</code>	104	<code>\zen@refitemText</code>	390, <u>400</u>
<code>\zen@Hytrue</code>	102	<code>\zen@refText</code>	104, 320, 335, 346
<code>\zen@ifrefundefined</code>	<u>65</u> , 324	<code>\zen@refText</code>	310
<code>\zen@info</code>	205, <u>460</u>	<code>\zen@refused</code>	325, <u>332</u>
<code>\zen@item@comma</code>	51, <u>504</u>	<code>\zen@setuprefs</code>	184, <u>196</u>
<code>\zen@item@comma@</code>	515, 519, 522, 524	<code>\zen@space</code>	375, 381, 389, 393
<code>\zen@itemizetrue</code>	42	<code>\zen@spaces</code>	365, 366, 369, 372
<code>\zen@keep</code>	292, 299, 300, 301	<code>\zen@starnamelist</code>	136, <u>153</u>
<code>\zen@keeplabel</code>	246, <u>292</u>	<code>\zen@temp</code>	71, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 115, 150, 155, 197, 198, 218, 238, 326, 327, 336, 337, 348, 349, 356, 373, 377, 378, 430, 431, 433
<code>\zen@labelize</code>	208, <u>242</u>	<code>\zen@toks</code>	<u>53</u> , 417, 418, 419, 420, 421, 433
<code>\zen@lastwarn</code>	322, 361	<code>\zen@trailingsp@ces</code>	118, 120, 124, 126
<code>\zen@list@anchors</code>	271, 280, 284, 285	<code>\zen@trailingspaces</code>	111, 118
<code>\zen@lowercase</code>	<u>61</u> , 232, 235, 323, 450	<code>\zen@trivlist</code>	180, <u>182</u>
<code>\zen@makelabel</code>	<u>196</u>	<code>\zen@trivlistORI</code>	179, 183
<code>\zen@makelabelORI</code>	199, 202, 208	<code>\zen@type</code>	141, <u>186</u>
<code>\zen@makeref</code>	208, <u>221</u> , 247	<code>\zen@unique</code>	<u>68</u>
<code>\zen@makeref@item@comma</code>	240, <u>248</u>	<code>\zen@warn</code>	345, 350, 361
<code>\zen@MultipleLabels</code>	337, <u>455</u>	<code>\zen@whichinfo</code>	462, <u>494</u>
<code>\zen@name@list</code>	181	<code>\zeninfo</code>	7, 205, <u>480</u>
<code>\zen@namelist</code>	135, <u>137</u> , 158	<code>\zeninfo@err</code>	482, 483, 484, 485, 489, 493
<code>\zen@nil</code>	418, 421	<code>\zenItem</code>	<u>53</u>
<code>\zen@norefitem</code>	331	<code>\zref@addprop</code>	96, 103
<code>\zen@Normalize</code>	<u>107</u> , 149, 154, 232, 235, 238, 323, 450	<code>\zref@extract</code>	317
<code>\zen@NormalizeCommands</code>	110, 130, 134	<code>\zref@extractdefault</code>	307, 348, 512, 513
<code>\zen@postsp@ce</code>	127, 128	<code>\zref@getcurrent</code>	95
<code>\zen@postspace</code>	123, 127	<code>\zref@labelbylist</code>	245
<code>\zen@propcount</code>	400, 438, 442	<code>\zref@labelbyprops</code>	244
<code>\zen@ref</code>	319	<code>\zref@newlist</code>	68
<code>\zen@ref@</code>	321, 335, 346, 351, 352, 354, 397, 451, 453	<code>\zref@newprop</code>	70, 72, 77
<code>\zen@refHy</code>	104, 311, 313, 351	<code>\zref@wrapper@unexpanded</code>	510
<code>\zen@refitem</code>	511, 514, 515, 516		
<code>\zen@refitem@</code>	254, 257, 261, 263		
<code>\zen@refitem@fun</code>	177, 215, 229, 231, 232		
<code>\zen@refitem@implicit</code>	206, <u>209</u>		
<code>\zen@refitem@noText</code>	394, 448		
<code>\zen@refitem@prop</code>	340, 359, <u>373</u>		
<code>\zen@refitemExternal</code>	321, <u>449</u>		
<code>\zen@refitemFromAnchor</code>	329, <u>342</u>		
<code>\zen@refitemfromlabel</code>	212, 217, 220		
<code>\zen@refitemOpt</code>	310, 311, <u>321</u>		
<code>\zen@refitemPrint</code>	380, 383, <u>386</u>		
<code>\zen@refitemProp</code>	373, 376, 385		

È

`\È` 19, 36

Ï

`\Ï` 17, 49

Ð

`\Ð` 18, 50