

# The zref package

Heiko Oberdiek  
<heiko.oberdiek at gmail.com>

2011/03/18 v2.21

## Abstract

Package `zref` tries to get rid of the restriction in  $\LaTeX$ 's reference system that only two properties are supported. The package implements an extensible referencing system, where properties are handled in a more flexible way. It offers an interface for macro programmers for the access to the system and some applications that uses the new reference scheme.

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Standard $\LaTeX$ behaviour	4
1.2	Basic idea	4
1.3	Interfaces	4
<b>2</b>	<b>Interface for programmers</b>	<b>5</b>
2.1	Entities	5
2.2	Property list	5
2.3	Property	6
2.4	Reference generation	6
2.5	Data extraction	7
2.6	Setup	8
2.7	Declared properties	9
2.8	Wrapper for advanced situations	9
2.9	Counter for unique names	10
<b>3</b>	<b>User interface</b>	<b>10</b>
3.1	Module <code>user</code>	10
3.2	Module <code>abspage</code>	11
3.3	Module <code>lastpage</code>	11
3.3.1	Tests for last page	12
3.3.2	Example	12
3.4	Module <code>thepage</code>	13
3.5	Module <code>nextpage</code>	13
3.5.1	Configuration	13
3.5.2	Example	14
3.6	Module <code>totpages</code>	14
3.7	Module <code>pagelayout</code>	15
3.8	Module <code>marks</code>	15
3.9	Module <code>runs</code>	15
3.10	Module <code>perpage</code>	16
3.11	Module <code>counter</code>	16
3.12	Module <code>titleref</code>	16
3.13	Module <code>savepos</code>	17
3.14	Module <code>dotfill</code>	18
3.15	Module <code>env</code>	18
3.16	Module <code>xr</code>	18

<b>4</b>	<b>ToDo</b>	<b>19</b>
<b>5</b>	<b>Example</b>	<b>19</b>
<b>6</b>	<b>Implementation</b>	<b>21</b>
6.1	Package <code>zref</code>	21
6.1.1	Identification	21
6.1.2	Load basic module	22
6.1.3	Process options	22
6.2	Module <code>base</code>	22
6.2.1	Prefixes	22
6.2.2	Identification	23
6.2.3	Utilities	23
6.2.4	Check for $\varepsilon$ -TeX	24
6.2.5	Auxiliary file stuff	24
6.2.6	Property lists	24
6.2.7	Properties	28
6.2.8	Reference generation	30
6.2.9	Reference querying and extracting	33
6.2.10	Compatibility with <code>babel</code>	36
6.2.11	Unique counter support	36
6.2.12	Utilities	37
6.2.13	Setup	37
6.3	Module <code>user</code>	38
6.4	Module <code>abspage</code>	39
6.5	Module <code>counter</code>	40
6.6	Module <code>lastpage</code>	40
6.7	Module <code>thepage</code>	41
6.8	Module <code>nextpage</code>	42
6.9	Module <code>totpages</code>	43
6.10	Module <code>pagelayout</code>	44
6.11	Module <code>pageattr</code>	47
6.12	Module <code>marks</code>	50
6.13	Module <code>runs</code>	51
6.14	Module <code>perpage</code>	51
6.15	Module <code>titleref</code>	53
6.15.1	Implementation	54
6.15.2	User interface	55
6.15.3	Patches for section and caption commands	55
6.15.4	Environment description	56
6.15.5	Class <code>memoir</code>	56
6.15.6	Class <code>beamer</code>	57
6.15.7	Package <code>titlesec</code>	57
6.15.8	Package <code>longtable</code>	58
6.15.9	Package <code>listings</code>	58
6.15.10	Theorems	58
6.16	Module <code>xr</code>	58
6.17	Module <code>hyperref</code>	66
6.18	Module <code>savepos</code>	66
6.18.1	Identification	66
6.18.2	Availability	66
6.18.3	Setup	67
6.18.4	User macros	67
6.19	Module <code>abspos</code>	68
6.19.1	Identification	68
6.20	Module <code>dotfill</code>	68
6.21	Module <code>env</code>	69

<b>7</b>	<b>Test</b>	<b>70</b>
7.1	<code>\zref@localaddprop</code> . . . . .	70
7.2	Module base . . . . .	70
7.3	Module runs . . . . .	71
7.4	Module titleref . . . . .	72
<b>8</b>	<b>Installation</b>	<b>73</b>
8.1	Download . . . . .	73
8.2	Bundle installation . . . . .	73
8.3	Package installation . . . . .	73
8.4	Refresh file name databases . . . . .	74
8.5	Some details for the interested . . . . .	74
<b>9</b>	<b>References</b>	<b>75</b>
<b>10</b>	<b>History</b>	<b>75</b>
	[2006/02/20 v1.0] . . . . .	75
	[2006/05/03 v1.1] . . . . .	75
	[2006/05/25 v1.2] . . . . .	75
	[2006/09/08 v1.3] . . . . .	75
	[2007/01/23 v1.4] . . . . .	75
	[2007/02/18 v1.5] . . . . .	76
	[2007/04/06 v1.6] . . . . .	76
	[2007/04/17 v1.7] . . . . .	76
	[2007/04/22 v1.8] . . . . .	76
	[2007/05/02 v1.9] . . . . .	76
	[2007/05/06 v2.0] . . . . .	76
	[2007/05/28 v2.1] . . . . .	76
	[2008/09/21 v2.2] . . . . .	76
	[2008/10/01 v2.3] . . . . .	76
	[2009/08/07 v2.4] . . . . .	76
	[2009/12/06 v2.5] . . . . .	76
	[2009/12/07 v2.6] . . . . .	77
	[2009/12/08 v2.7] . . . . .	77
	[2010/03/26 v2.8] . . . . .	77
	[2010/03/29 v2.9] . . . . .	77
	[2010/04/08 v2.10] . . . . .	77
	[2010/04/15 v2.11] . . . . .	77
	[2010/04/17 v2.12] . . . . .	77
	[2010/04/19 v2.13] . . . . .	77
	[2010/04/22 v2.14] . . . . .	78
	[2010/04/23 v2.15] . . . . .	78
	[2010/04/28 v2.16] . . . . .	78
	[2010/05/01 v2.17] . . . . .	78
	[2010/05/13 v2.18] . . . . .	78
	[2010/10/22 v2.19] . . . . .	78
	[2011/02/12 v2.20] . . . . .	79
	[2011/03/18 v2.21] . . . . .	79
<b>11</b>	<b>Index</b>	<b>79</b>

## 1 Introduction

Standard L<sup>A</sup>T<sub>E</sub>X's reference system with `\label`, `\ref`, and `\pageref` supports two properties, the appearance of the counter that is last incremented by `\refstepcounter` and the page with the `\label` command.

Unhappily L<sup>A</sup>T<sub>E</sub>X does not provide an interface for adding another properties. Packages such as `hyperref`, `nameref`, or `titleref` are forced to use ugly hacks to extend the reference system. These ugly hacks are one of the causes for `hyperref`'s difficulty regarding compatibility with other packages.

## 1.1 Standard L<sup>A</sup>T<sub>E</sub>X behaviour

References are created by the `\label` command:

```
\chapter{Second chapter}
\section{First section on page 7} % section 2.1
\label{myref}
```

Now L<sup>A</sup>T<sub>E</sub>X records the section number 2.1 and the page 7 in the reference. Internally the reference is a list with two entries:

```
\r@myref → {2.1}{7}
```

The length of the list is fixed in the L<sup>A</sup>T<sub>E</sub>X kernel. An interface for adding new properties is missing.

There are several tries to add new properties:

**hyperref** uses a list of five properties instead of the standard list with two entries.

This causes many compatibility problems with L<sup>A</sup>T<sub>E</sub>X and other packages.

**titleref** stores its title data into the first entry in the list. L<sup>A</sup>T<sub>E</sub>X is happy because it does only see its list with two entries. The situation becomes more difficult, if more properties are added this way. Then the macros form a nested structure inside the first reference argument for the label. Expandable extractions will then become painful.

## 1.2 Basic idea

Some time ago Morten Høgholm sent me an experimental cross referencing mechanism as “expl3” code. His idea is:

```
\g_xref_mylabel_plist →
  \xref_dance_key{salsa}\xref_name_key{Morten}...
```

The entries have the following format:

```
\xref_{your key}_key{some text}
```

This approach is much more flexible:

- New properties can easily be added, just use a new key.
- The length of the list is not fixed. A reference can use a subset of the keys.
- The order of the entries does not matter.

Unhappily I am not familiar with the experimental code for L<sup>A</sup>T<sub>E</sub>X3 that will need some time before its first release. Thus I have implemented it as L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> package without disturbing the existing L<sup>A</sup>T<sub>E</sub>X reference system.

## 1.3 Interfaces

The package provides a generic *interface for programmers*. Commands of this interface are prefixed by `\zref@`.

Option `user` enables the *user interface*. Here the commands are prefixed by `\z` to avoid name clashes with existing macros.

Then the package provides some *modules*. They are applications for the reference system and can also be considered as examples how to use the reference system.

The modules can be loaded as packages. The package name is prefixed with `zref-`, for example:

```
\RequirePackage{zref-abspage}
```

This is the preferred way if the package is loaded from within other packages to avoid option clashes.

As alternative package `zref` can be used and the modules are given as options:

```
\usepackage[perpage,user]{zref}
```

## 2 Interface for programmers

The user interface is described in the next section [3](#).

### 2.1 Entities

**Reference.** Internally a reference is a list of key value pairs:

```
\Z@R@myref → \default{2.1}\page{7}
```

The generic format of a entry is:

```
\Z@R@⟨refname⟩ → \⟨propname⟩{⟨value⟩}
```

⟨*refname*⟩ is the name that denoted references (the name used in `\label` and `\ref`). ⟨*propname*⟩ is the name of the property or key. The property key macro is never executed, it is used in parameter text matching only.

**Property.** Because the name of a property is used in a macro name that must survive the `.aux` file, the name is restricted to letters and '@'.

**Property list.** Often references are used for special purposes. Thus it saves memory if just the properties are used in this reference that are necessary for its purpose.

Therefore this package uses the concept of *property lists*. A property list is a set of properties. The set of properties that is used by the default `\label` command is the *main property list*.

### 2.2 Property list

<sup>exp</sup> means that the implementation of the marked macro is expandable. <sup>exp2</sup> goes a step further and marks the macro expandable in exact two expansion steps.

```
\zref@newlist {⟨listname⟩}
```

Declares a new empty property list.

```
\zref@addprop {⟨listname⟩} {⟨propname list⟩}
```

Adds the properties of ⟨*propname list*⟩ (comma separated) to the property list ⟨*listname*⟩. The property and list must exist. A ⟨*propname list*⟩ can be given since 2010/04/19 v2.13. Before this version only one property name could be added in one call of `\zref@addprop`.

```
\zref@localaddprop {⟨listname⟩} {⟨propname list⟩}
```

Local variant of `\zref@addprop`.

```
\zref@listexists {<listname>} {<then>}
```

Executes *<then>* if the property list *<listname>* exists or raise an error otherwise.

```
\zref@iflistundefinedexp {<listname>} {<then>} {<else>}
```

Executes *<then>* if the list exists or *<else>* otherwise.

```
\zref@iflistcontainsprop {<listname>} {<propname>} {<then>} {<else>}
```

Executes *<then>* if the property *<propname>* is part of property list *<listname>* or otherwise it runs the *<else>* part.

## 2.3 Property

```
\zref@newprop* {<propname>} [<default>] {<value>}
```

This command declares and configures a new property with name *<propname>*.

In case of unknown references or the property does not exist in the reference, the *<default>* is used as value. If it is not specified here, a global default is used, see `\zref@setdefault`.

The correct values of some properties are not known immediately but at page shipout time. Prominent example is the page number. These properties are declared with the star form of the command.

```
\zref@setcurrent {<propname>} {<value>}
```

This sets the current value of the property *<propname>*. It is a generalization of setting L<sup>A</sup>T<sub>E</sub>X's `\currentlabel`.

```
\zref@getcurrentexp2 {<propname>}
```

This returns the current value of the property *<propname>*. The value may not be correct, especially if the property is bound to a page (start form of `\zref@newprop`) and the right value is only known at shipout time (e.g. property 'page'). In case of errors (e.g. unknown property) the empty string is returned.

Since version 2010/04/22 v2.14 `\zref@getcurrent` supports `\zref@wrapper@unexpanded`.

```
\zref@propexists {<propname>} {<then>}
```

Calls *<then>* if the property *<propname>* is available or generates an error message otherwise.

```
\zref@ifpropundefinedexp {<propname>} {<then>} {<else>}
```

Calls *<then>* or *<else>* depending on the existence of property *<propname>*.

## 2.4 Reference generation

```
\zref@label {<refname>}
```

This works similar to `\label`. The reference *<refname>* is created and put into the `.aux` file with the properties of the main property list.

`\zref@labelbylist {<refname>} {<listname>}`

Same as `\zref@label` except that the properties are taken from the specified property list `<listname>`.

`\zref@labelbyprops {<refname>} {<propnameA>,<propnameB>,...}`

Same as `\zref@label` except that these properties are used that are given as comma separated list in the second argument.

`\zref@newlabel {<refname>} {...}`

This is the macro that is used in the `.aux` file. It is basically the same as `\newlabel` apart from the format of the data in the second argument.

## 2.5 Data extraction

`\zref@extractdefaultexp2 {<refname>} {<propname>} {<default>}`

This is the basic command that references the value of a property `<propname>` for the reference `<refname>`. In case of errors such as undefined reference the `<default>` is used instead.

`\zref@extractexp2 {<refname>} {<propname>}`

The command is an abbreviation for `\zref@extractdefault`. As default the default of the property is taken, otherwise the global default.

Example for page references:

```
LaTeX: \pageref{foobar}
zref:   \zref@extract{foobar}{page}
```

Both `\zref@extract` and `\zref@extractdefault` are expandable. That means, these macros can directly be used in expandable calculations, see the example file. On the other side, babel's shorthands are not supported, there are no warnings in case of undefined references.

If an user interface doesn't need expandable macros then it can use `\zref@refused` and `\zref@wrapper@babel` for its user macros.

`\zref@refused {<refname>}`

This command is not expandable. It causes the warnings if the reference `<refname>` is not defined. Use the `\zref@extract` commands inside expandable contexts and mark their use outside by `\zref@refused`, see the example file.

`\zref@ifrefundefinedexp {<refname>} {<then>} {<else>}`

A possibility to check whether a reference exists.

`\zifrefundefined {<refname>} {<then>} {<else>}`

Macro `\zifrefundefined` calls `\zref@refused` before executing `\zref@ifrefundefined`. Babel shorthands are supported in `<refname>`.

`\zref@ifrefcontainspropexp {<refname>} {<propname>} {<then>} {<else>}`

Test whether a reference provides a property.

## 2.6 Setup

`\zref@default`

Holds the global default for unknown values.

`\zref@setdefault {<value>}`

Sets the global default for unknown values. The global default is used, if a property does not specify an own default and the value for a property cannot be extracted. This can happen if the reference is unknown or the reference does not have the property.

`\zref@setmainlist {<value>}`

Sets the name of the main property list. The package sets and uses `main`.



## 2.7 Declared properties

Module	Property	Property list	Default
(base)	default	main	<code>&lt;empty&gt;</code>
	page	main	<code>&lt;empty&gt;</code>
abspage	abspage	main	0
counter	counter	main	<code>&lt;empty&gt;</code>
hyperref	anchor	main	<code>&lt;empty&gt;</code>
	url		<code>&lt;empty&gt;</code>
pageattr	pdfpageattr	thepage	...
	pdfpagesattr	LastPage	...
pagelayout <sup>1</sup>	mag	thepage	<code>\number\mag</code>
	paperwidth	thepage	<code>\number\paperwidth</code>
	paperheight	thepage	<code>\number\paperheight</code>
	stockwidth	thepage	<code>\number\stockwidth</code>
	stockheight	thepage	<code>\number\stockheight</code>
	pdfpageheight	thepage	<code>\number\pdfpageheight</code>
	pdfpagewidth	thepage	<code>\number\pdfpagewidth</code>
	pdfhorigin	thepage	<code>\number\pdfhorigin</code>
	pdfvorigin	thepage	<code>\number\pdfvorigin</code>
	hoffset	thepage	<code>\number\hoffset</code>
	voffset	thepage	<code>\number\voffset</code>
	topmargin	thepage	<code>\number\topmargin</code>
	oddsidemargin	thepage	<code>\number\oddsidemargin</code>
	evensidemargin	thepage	<code>\number\evensidemargin</code>
	textwidth	thepage	<code>\number\textwidth</code>
	textheight	thepage	<code>\number\textheight</code>
	headheight	thepage	<code>\number\headheight</code>
	headsep	thepage	<code>\number\headsep</code>
	footskip	thepage	<code>\number\footskip</code>
	marginparwidth	thepage	<code>\number\marginparwidth</code>
marginparsep	thepage	<code>\number\marginparsep</code>	
columnwidth	thepage	<code>\number\columnwidth</code>	
columnsep	thepage	<code>\number\columnsep</code>	
perpage	pagevalue	perpage	0
	page	perpage	<code>&lt;empty&gt;</code>
	abspage	perpage	0
savepos	posx	savepos	0
	posy	savepos	0
titleref	title	main	<code>&lt;empty&gt;</code>
xr	anchor		<code>&lt;empty&gt;</code>
	externaldocument		<code>&lt;empty&gt;</code>
	thetype		<code>&lt;empty&gt;</code>
	title		<code>&lt;empty&gt;</code>
	url		<code>&lt;empty&gt;</code>

## 2.8 Wrapper for advanced situations

```
\zref@wrapper@babel {...} {\langle name \rangle}
```

This macro helps to add shorthand support. The second argument is protected, then the code of the first argument is called with the protected name appended. Examples are in the sources.

<sup>1</sup>Module `pagelayout` only defines properties if the parameter exists.

`\zref@wrapper@immediate {...}`

There are situations where a label must be written instantly to the `.aux` file, for example after the last page. If the `\zlabel` or `\label` command is put inside this wrapper, immediate writing is enabled. See the implementation for module `lastpage` for an example of its use.

`\zref@wrapper@unexpanded {...}`

Assuming someone wants to extract a value for property `bar` and store the result in a macro `\foo` without traces of the expanding macros and without expanding the value. This (theoretical?) problem can be solved by this wrapper:

```
\zref@wrapper@unexpanded{%
  \edef\foo{%
    \zref@extract{someref}{bar}%
  }%
}
```

The `\edef` forces the expansion of `\zref@extract`, but the extraction of the value is prevented by the wrapper that uses  $\varepsilon$ -TeX' `\unexpanded` for this purpose. Supported macros are `\zref@extract`, `\zref@extractdefault` and since version 2010/04/22 v2.14 macro `\zref@getcurrent`.

## 2.9 Counter for unique names

Some modules (`titleref` and `dotfillmin`) need unique names for automatically generated label names.

`\zref@require@unique`

This command creates the unique counter `zref@unique` if the counter does not already exist.

`\thezref@unique`

This command is used to generate unique label names.

## 3 User interface

### 3.1 Module user

The user interface for this package and its modules is enabled by `zref`'s package option `user` or package `zref-user`. The names of user commands are prefixed by `z` in order to avoid name clashes with existing macros of the same functionality. Thus the package does not disturb the traditional reference scheme, both can be used together.

The syntax descriptions contain the following markers that are intended as hints for programmers:

<code>babel</code>	Babel shorthands are allowed.
<code>robust</code>	Robust macro.
<code>exp</code>	Expandable version: <ul style="list-style-type: none"><li>• robust, unless the extracted values are fragile,</li><li>• no babel shorthand support.</li></ul>
<code>exp2</code>	Expandable like <code>exp</code> and: <ul style="list-style-type: none"><li>• expandable in exact two steps.</li></ul>

The basic user interface of the package without modules are commands that mimic the standard L<sup>A</sup>T<sub>E</sub>X behaviour of `\label`, `\ref`, and `\pageref`:

```
\zlabel {⟨refname⟩}babel
```

Similar to `\label`. It generates a label with name `⟨refname⟩` in the new reference scheme.

```
\zref [⟨propname⟩] {⟨refname⟩}babel
```

Without optional argument similar to `\ref`, it returns the default reference property. This property is named `default`:

```
\zref{x} ≡ \zref[default]{x}
```

```
\zpageref {⟨refname⟩}babel
```

Convenience macro, similar to `\pageref`.

```
\zpageref{x} ≡ \zref[page]{x}
```

```
\zrefused {⟨refname⟩}babel
```

Some of the user commands in the modules are expandable. The use of such commands do not cause any undefined reference warnings, because inside of expandable contexts this is not possible. However, if there is a place outside of expandable contexts, `\zrefused` is strongly recommended. The reference `⟨refname⟩` is marked as used, undefined ones will generate warnings.

### 3.2 Module `abspage`

With the help of package `atbegshi` a new counter `abspage` with absolute page numbers is provided. Also a new property `abspage` is defined and added to the main property list. Thus you can reference the absolute page number:

```
Section \zref{foo} is on page \zpageref{foo}.
This is page \zref[abspage]{foo}
of \zref[abspage]{LastPage}.
```

The example also makes use of module `lastpage`.

### 3.3 Module `lastpage`

Provides the functionality of package `lastpage` [3] in the new reference scheme. The label `LastPage` is put at the end of the document. You can refer the last page number with:

```
\zref@extract{LastPage}{page} (+ \zref@refused{LastPage})
```

or

```
\zpageref{LastPage} (module user)
```

Since version 2008/10/01 v2.3 the module defines the list `LastPage`. In addition to the properties of the main list label `LastPage` also stores the properties of this list `LastPage`. The default of this list is empty. The list can be used by the user to add additional properties for label `LastPage`.

### 3.3.1 Tests for last page

Since version 2010/03/26 v2.8 the macros `\zref@iflastpage` and `\ziflastpage` were added. They test the reference, whether it is a reference of the last page.

```
\zref@iflastpageexp {<refname>} {<then>} {<else>}
```

Macro `\zref@iflastpage` compares the references `<refname>` with `<LastPage>`. Basis of the comparison is the value of property `abspage`, because the values are different for different pages. This is not ensured by property `page`. Therefore module `abspage` is loaded by module `lastpage`. If both values of property `abspage` are present and match, then `<then>` is executed, otherwise code `<else>` is called. If one or both references are undefined or lack the property `abspage`, then `<else>` is executed.

Macro `\zref@iflastpage` is expandable, therefore `\zref@refused` should be called on `<refname>` and `<LastPage>`.

```
\ziflastpage {<refname>} {<then>} {<else>}
```

Macro `\ziflastpage` has the same function as `\zref@iflastpage`, but adds support for babel shorthands in `<refname>` and calls `\zref@refused`. However macro `\ziflastpage` is not expandable.

### 3.3.2 Example

```
1 *example-lastpage
2 %<END_EXAMPLE
3 \NeedsTeXFormat{LaTeX2e}
4 \documentclass{report}
5
6 \newcounter{foo}
7 \renewcommand*{\thefoo}{\Alph{foo}}
8
9 \usepackage{zref-lastpage,zref-user}[2011/03/18]
10
11 \makeatletter
12 \zref@newprop{thefoo}{\thefoo}
13 \zref@newprop{valuefoo}{\the\value{foo}}
14 \zref@newprop{chapter}{\thechapter}
15 \zref@addprops{LastPage}{thefoo,valuefoo,chapter}
16 \makeatother
17
18 \newcommand*{\foo}{%
19   \stepcounter{foo}%
20   [Current foo: \thefoo]%
21 }
22
23 \begin{document}
24   \chapter{First chapter}
25   Last page is \zref{LastPage}.\
26   Last chapter is \zref[chapter]{LastPage}.\
27   Last foo is \zref[thefoo]{LastPage}.\
28   Last value of foo is \zref[valuefoo]{LastPage}.\
29   \foo
30   \chapter{Second chapter}
31   \foo\foo\foo
32   \chapter{Last chapter}
33   \foo
34 \end{document}
35 %END_EXAMPLE
```

### 3.4 Module `thepage`

This module `thepage` loads module `abspage`, constructs a reference name using the absolute page number and remembers property `page`. Other properties can be added by adding them to the property list `thepage`.

```
\zthepage {<absolute page number>}
```

Macro `\zthepage` is basically a `\zpageref`. The reference name is yield by the `<absolute page number>`. If the reference is not defined, then the default for property `page` is used.

```
\zref@thepage@nameexp {<absolute page number>}
```

Macro `\zref@thepage@name` returns the internal reference name that is constructed using the `<absolute page number>`. The internal reference name should not be used directly, because it might change in future versions.

```
\zref@thepageexp {<absolute page number>}
\zref@thepage@refused {<absolute page number>}
```

Macro `\zref@thepage` returns the page number (`\thepage`) of `<absolute page number>`. Because this macro is expandable, `\zref@thepage@refused` is used outside an expandable context to mark the reference as used.

### 3.5 Module `nextpage`

```
\znexpage
```

Macro `\znexpage` prints `\thepage` of the following page. It gets the current absolute page number by using a label. There are three cases for the next page:

1. The next page is not known yet because of undefined references. Then `\zunknownnextpagename` is used instead. The default for this macro is the default of property `page`.
2. This page is the last page. Then `\znonextpagename` is used. Its default is empty.
3. The next page is known, then `\thepage` of the next page is used (the value of property `page` of the next page).

#### 3.5.1 Configuration

The behaviour can be configured by the following macros.

```
\zunknownnextpagename
\znonextpagename
```

If the next page is not known or available, then `\znexpage` uses these name macros as default. `\zunknownnextpagename` is used in case of undefined references. Default is the value of property `page` of the next page (`\thepage`). Module `thepage` is used.

Macro `\znonextpagename` is used, if the next page does not exist. That means that the current page is the last page. The default is empty.

<code>\znextpagesetup {&lt;unknown&gt;} {&lt;no next&gt;} {&lt;next&gt;}</code>
---------------------------------------------------------------------------------

According to the case (see `\znextpage`) macro `\znextpage` calls an internal macro with an argument. The argument is either `\thepage` of the next page or one of `\zunknownnextpagename` or `\znonextpagename`. These internal macros can be changed by `\znextpagesetup`. It expects the definition texts for these three cases of a macro with one argument. The default is

```
\znextpagesetup{#1}{#1}{#1}
```

### 3.5.2 Example

```
37 <*example-nextpage>
38 %<<END_EXAMPLE
39 \documentclass{book}
40
41 \usepackage{zref-nextpage}[2011/03/18]
42 \znextpagesetup
43   {\thepage}% next page is unknown
44   {\thepage\ (#1)}% this page is last page
45   {\thepage\ $\rightarrow$ #1}% next page is known
46 \renewcommand*{\znonextpagename}{last page}
47
48 \usepackage{fancyhdr}
49 \pagestyle{fancy}
50 \fancyhf{}
51 \fancyhead[LE,RO]{\znextpage}
52 \fancypagestyle{plain}{%
53   \fancyhf{}%
54   \fancyhead[LE,RO]{\znextpage}%
55 }
56
57 \begin{document}
58 \frontmatter
59 \tableofcontents
60 \mainmatter
61 \chapter{Hello World}
62 \clearpage
63 \section{Last section}
64 \end{document}
65 %END_EXAMPLE
66 </example-nextpage>
```

## 3.6 Module `totpages`

For the total number of pages of a document you need to know the absolute page number of the last page. Both modules `abspage` and `lastpage` are necessary and automatically enabled.

<code>\ztotpages<sup>exp</sup></code>
---------------------------------------

Prints the total number of pages or 0 if this number is not yet known. It expands to an explicit number and can also be used even in expandable calculations (`\numexpr`) or counter assignments.

### 3.7 Module pagelayout

The module defines additional properties for each parameter of the page layout that is effective during page shipout. The value of length parameters is given in sp without the unit as plain number.

Some parameters are specific for a class (e.g. `stockwidth` and `stockheight` for class `memoir`) or the  $\text{\TeX}$  engine like `pdf $\text{\TeX}$` . If the parameter is not available, then the property will not be defined. The default value of the property is the current setting of the parameter.

The module `thepage` is loaded that generates a label for each page. The properties of module `pagelayout` are added to the property list `thepage` of module `thepage`.

List of properties:

parameter	property	remarks
<code>\mag</code>	<code>mag</code>	
<code>\paperwidth</code>	<code>paperwidth</code>	
<code>\paperheight</code>	<code>paperheight</code>	
<code>\stockwidth</code>	<code>stockwidth</code>	class <code>memoir</code>
<code>\stockheight</code>	<code>stockheight</code>	class <code>memoir</code>
<code>\pdfpagewidth</code>	<code>pdfpagewidth</code>	<code>pdf<math>\text{\TeX}</math></code> , <code>Lua<math>\text{\TeX}</math></code>
<code>\pdfpageheight</code>	<code>pdfpageheight</code>	<code>pdf<math>\text{\TeX}</math></code> , <code>Lua<math>\text{\TeX}</math></code>
<code>\pdfhorigin</code>	<code>pdfhorigin</code>	<code>pdf<math>\text{\TeX}</math></code> , <code>Lua<math>\text{\TeX}</math></code>
<code>\pdfvorigin</code>	<code>pdfvorigin</code>	<code>pdf<math>\text{\TeX}</math></code> , <code>Lua<math>\text{\TeX}</math></code>
<code>\hoffset</code>	<code>hoffset</code>	
<code>\voffset</code>	<code>voffset</code>	
<code>\topmargin</code>	<code>topmargin</code>	
<code>\oddsidemargin</code>	<code>oddsidemargin</code>	
<code>\evensidemargin</code>	<code>evensidemargin</code>	
<code>\textwidth</code>	<code>textwidth</code>	
<code>\textheight</code>	<code>textheight</code>	
<code>\headheight</code>	<code>headheight</code>	
<code>\headsep</code>	<code>headsep</code>	
<code>\footskip</code>	<code>footskip</code>	
<code>\marginparwidth</code>	<code>marginparwidth</code>	
<code>\marginparsep</code>	<code>marginparsep</code>	
<code>\columnwidth</code>	<code>columnwidth</code>	
<code>\columnsep</code>	<code>columnsep</code>	

`\zlistpagelayout`

At the end of document the page layout parameter for each page are printed into the `.log` file if macro `\zlistpagelayout` is called before `\end{document}` (preamble is a good place).

### 3.8 Module marks

ToDo.

### 3.9 Module runs

Module `runs` counts the  $\text{\LaTeX}$  runs since last `.aux` file creation and prints the number in the `.log` file.

`\zrunsexp`

Prints the the total number of  $\text{\LaTeX}$  runs including the current one. It expands to an explicit number. Before `begin{document}` the value is zero meaning the `.aux`

file is not read yet. If a previous `.aux` file exists, the value found there increased by one is the new number. Otherwise `\zruns` is set to one. L<sup>A</sup>T<sub>E</sub>X runs where the `.aux` files are not rewritten are not counted (see `\nofiles`).

### 3.10 Module `perpage`

With `\@addtoreset` or `\numberwithin` a counter can be reset if another counter is incremented. This do not work well if the other counter is the page counter. The page counter is incremented in the output routine that is often called asynchronous somewhere on the next page. A reference mechanism costs at least two L<sup>A</sup>T<sub>E</sub>X runs, but ensures correct page counter values.

```
\zmakeperpage [reset] {counter}
```

At the of a new page counter `<counter>` starts counting with value `<reset>` (default is 1). The macro has the same syntax and semantics as `\MakePerPage` of package `perpage` [5]. Also `perpage` of package `footmisc` [1] can easily be simulated by

```
\zmakeperpage{footnote} % \usepackage[perpage]{footmisc}
```

If footnote symbols are used, some people dislike the first symbol †. It can easily be skipped:

```
\zmakeperpage[2]{footnote}
```

```
\thezpage
counter zpage
```

If the formatted counter value of the counter that is reset at a new page contains the page value, then you can use `\thezpage`, the page number of the current page. Or counter `zpage` can be used, if the page number should be formatted differently from the current page number. Example:

```
\newcounter{foobar}
\zmakeperpage{foobar}
\renewcommand*\thefoobar{\thezpage-\arabic{foobar}}
% or
\renewcommand*\thefoobar{\roman{zpage}-\arabic{foobar}}
```

```
\zunmakeperpage {counter}
```

The reset mechanism for this counter is deactivated.

### 3.11 Module `counter`

This option just add the property `counter` to the main property list. The property stores the counter name, that was responsible for the reference. This is the property `hyperref`'s `\autoref` feature uses. Thus this property `counter` may be useful for a reimplementaion of the `autoref` feature, see the section 4 with the `todo` list.

### 3.12 Module `titleref`

This option makes section and caption titles available to the reference system similar to packages `titleref` or `nameref`.



`\ztitleref {⟨refname⟩}^babel`

Print the section or caption title of reference  $\langle refname \rangle$ , similar to `\nameref` or `\titleref`.

`\ztitlerefsetup {key1=value1, key2=value2, ...}`

This command allows to configure the behaviour of module `titleref`. The following keys are available:

`title=⟨value⟩`

Sets the current title.

`stripperiod=true|false`

Follow package `nameref` that removes a last period. Default: `true`.

`expand=true|false`

Package `\titleref` expands the title first. This way garbage and dangerous commands can be removed, e.g. `\label`, `\index...`. See implementation section for more details. Default is `false`.

`cleanup={...}`

Hook to add own cleanup code, if method `expand` is used. See implementation section for more details.

### 3.13 Module `savepos`

This option supports a feature that `pdfTeX` provides (and `XYTeX`). `pdfTeX` is able to tell the current position on the page. The page position is not instantly known. First the page must be constructed by `TeX`'s asynchronous output routine. Thus the time where the position is known is the page shipout time. Thus a reference system where the information is recorded in the first run and made available for use in the second run comes in handy.

`\zsavepos {⟨refname⟩}`

It generates a reference with name  $\langle refname \rangle$ . The reference stores the location where `\zsavepos` is executed in properties `posx` and `posy`.

`\zposxexp {⟨refname⟩}`  
`\zposyexp {⟨refname⟩}`

Get the position as number. Unit is sp. Horizontal positions by `\zposx` increase from left to right. Vertical positions by `\zposy` from bottom to top.

Do not rely on absolute page numbers. Because of problems with the origin the numbers may differ in DVI or PDF mode of `pdfTeX`. Therefore work with relative values by comparisons.

Both `\zposx` and `\zposy` are expandable and can be used inside calculations (`\setcounter`, `\addtocounter`, package `calc`, `\numexpr`). However this property prevents from notifying `LATEX` that the reference is actually used (the notifying is not expandable). Therefore you should mark the reference as used by `\zrefused`.

This module uses `pdfTeX`'s `\pdfsavepos`, `\pdflastxpos`, and `\pdflastypos`. They are available in PDF mode and since version 1.40.0 also in DVI mode.

`\zref@savepos`

Macro `\zref@savepos` performs the first part of `\zsavepos` by calling `\pdfsavepos` (if `.aux` files are writable).

Thus `\zsavepos` is basically `\zref@savepos` followed by `\zreflabelbylist{⟨refname⟩}{savepos}`. The property list `savepos` contains the properties `posx` and `posy`.

### 3.14 Module `dotfill`

`\zdotfill`

This package provides the command `\zdotfill` that works similar to `\dotfill`, but can be configured. Especially it suppresses the dots if a minimum number of dots cannot be set.

`\zdotfillsetup {key1=value1, key2=value2, ...}`

This command allows to configure the behaviour of `\zdotfill`. The following keys are available:

`min=⟨count value⟩`

If the actual number of dots are smaller than `⟨count value⟩`, then the dots are suppressed. Default: 2.

`unit=⟨dimen value⟩`

The width of a dot unit is given by `⟨dimen value⟩`. Default: 0.44em (same as the unit in `\dotfill`).

`dot=⟨value⟩`

The dot itself is given by `⟨value⟩`. Default: `.` (dot, same as the dot in `\dotfill`).

### 3.15 Module `env`

This module defines two properties `envname` and `envline`. They remember the name of the environment and the line number at the start of the environment.

### 3.16 Module `xr`

This package provides the functionality of package `xr`, see [8]. It also supports the syntax of `xr-hyper`.

`\zexternaldocument * [⟨prefix⟩]babel {⟨external document⟩} [⟨url⟩]`

See `\externaldocument` for a description of this option. The found labels also get a property `externaldocument` that remembers `⟨external document⟩`. The standard reference scheme and the scheme of this package use different name spaces for reference names. If the external document uses both systems. Then one import statement would put the names in one namespace and probably causing problems with multiple references of the same name. Thus the star form only looks for `\newlabel` in the `.aux` files, whereas without star only `\zref@newlabels` are used.

In the star form it tries to detect labels from `hyperref`, `titleref`, and `ntheorem`. If such an extended property from the packages before cannot be found or are empty, they are not included in the imported reference.

Warnings are given if a reference name is already in use and the item is ignored. Unknown properties will automatically be declared.

If the external references contain `anchor` properties, then we need also a url to be able to address the external file. As default the filename is taken with a default extension.

```
\zxrsetup {key1=value1, key2=value2, ...}
```

The following setup options are available:

**ext:** It sets the default extension.

**tozreflabel:** Boolean option. The found references are imported as zref labels. This is enabled by default.

**toitxlabel:** Boolean option. The found references are imported as L<sup>A</sup>T<sub>E</sub>X labels. Packages `nameref`, `titleref` and class `memoir` are supported.

**urluse:** Boolean option. If enabled, then a URL is stored in a macro and the macro is put in property ‘urluse’. The URL is not put in property ‘url’. The purpose is to save T<sub>E</sub>X memory.

**verbose:** Boolean option. List the imported labels in the `.log` file. Default is `false`.

```
\zref@xr@ext
```

If the `<url>` is not specified in `\zref@externaldocument`, then the url will be constructed with the file name and this macro as extension. `\XR@ext` is used if `hyperref` is loaded, otherwise `pdf`.

## 4 ToDo

Among other things the following issues are left for future work:

- Other applications: `autoref`, `hyperref`, ...

## 5 Example

```
67 <*example>
68 \documentclass{book}
69
70 \usepackage[ngerman]{babel}%
71
72 \usepackage[savepos,totpages,titleref,dotfill,counter,user]{zref}
73
```

Chapters are wrapped inside `\ChapterStart` and `\ChapterStop`. The first argument `#1` of `\ChapterStart` is used to form a label id `chap:#1`. At the end of the chapter another label is set by `\zref@wrapper@immediate`, because otherwise at the end of document a deferred write would not be written, because there is no page for shipout.

Also this example shows how chapter titles can be recorded. A new property `chaptitle` is declared and added to the main property list. In `\ChapterStart` the current value of the property is updated.

```
74 \makeatletter
75 \zref@newprop{chaptitle}{}
76 \zref@addprop{main}{chaptitle}
77
78 \newcommand*{\ChapterStart}[2]{%
79   \cleardoublepage
80   \def\current@chapid{#1}%
81   \zref@setcurrent{chaptitle}{#2}%
82   \chapter{#2}%
83   \zlabel{chap:#1}%
84 }
```

```

85 \newcommand*\ChapterStop}{%
86   \cleardoublepage
87   \zref@wrapper@immediate{%
88     \zref@labelbyprops{chapend:\current@chapid}{abspage}%
89   }%
90 }

```

`\ChapterPages` calculates and returns the number of pages of the referenced chapter.

```

91 \newcommand*\ChapterPages}[1]{%
92   \zrefused{chap:#1}%
93   \zrefused{chapend:#1}%
94   \number\numexpr
95     \zref@extract{chapend:#1}{abspage}%
96     -\zref@extract{chap:#1}{abspage}%
97     +1\relax
98 }
99 \makeatother
100 \begin{document}

```

As exception we use `\makeatletter` here, because this is just an example file that also should show some of programmer's interface.

```

101 \makeatletter
102
103 \frontmatter
104 \zlabel{documentstart}
105
106 \begin{itemize}
107 \item
108   The frontmatter part has
109   \number\numexpr\zref@extract{chap:first}{abspage}-1\relax
110   ~pages.
111 \item
112   Chapter \zref{chap:first} has \ChapterPages{first} page(s).
113 \item
114   Section \zref{hello} is on the
115   \ifcase\numexpr
116     \zref@extractdefault{hello}{page}{0}%
117     -\zref@extractdefault{chap:first}{page}{0}%
118     +1\relax
119     ??\or first\or second\or third\or forth\fi
120   ~page inside its chapter.
121 \item
122   The document has
123   \zref[abspage]{LastPage} pages.
124   This number is \ifodd\ztotpages odd\else even\fi.
125 \item
126   The last page is labeled with \pageref{LastPage}.
127 \item
128   The title of chapter \zref{chap:next} %
129   is ‘‘\zref[chaptitle]{chap:next}’’.
130 \end{itemize}
131
132 \tableofcontents
133
134 \mainmatter
135 \ChapterStart{first}{First chapter}
136

```

The user level commands should protect babel shorthands where possible. On the other side, expandable extracting macros are useful in calculations, see above the examples with `\numexpr`.

```

137 \section{Test}
138 \zlabel{a"o}

```

```

139 Section \zref{a"o} on page
140 \zref@wrapper@babel\zref@extract{a"o}{page}.
141
142 Text.
143 \newpage
144
145 \section{Hello World}
146 \zlabel{hello}
147
148 \ChapterStop
149
150 \ChapterStart{next}{Next chapter with \emph{umlauts}: "a"o"u"s}
151

```

Here an example follows that makes use of pdfTeX's "savepos" feature. The position on the page is not known before the page is constructed and shipped out. Therefore the position is stored in references and are available for calculations in the next L<sup>A</sup>T<sub>E</sub>X compile run.

```

152 The width of the first column is
153 \the\dimexpr \zposx{secondcol}sp - \zposx{firstcol}sp\relax,\!
154 the height difference of the two baselines is
155 \the\dimexpr \zposy{firstcol}sp - \zposy{secondline}sp\relax:\!
156 \begin{tabular}{ll}
157 \zsavepos{firstcol}Hello&\zsavepos{secondcol}World\!
158 \zsavepos{secondline}Second line&foobar\!
159 \end{tabular}
160

```

With \zrefused L<sup>A</sup>T<sub>E</sub>X is notified, if the references are not yet available and L<sup>A</sup>T<sub>E</sub>X can generate the rerun hint.

```

161 \zrefused{firstcol}
162 \zrefused{secondcol}
163 \zrefused{secondline}
164
165 \ChapterStop
Test for module \dotfill.
166 \ChapterStart{dotfill}{Test for dotfill feature}
167 \newcommand*{\dfctest}[1]{%
168 #1&
169 [\makebox[#{#1}]{\dotfill}]&
170 [\makebox[#{#1}]{\zdotfill}]\!
171 }
172 \begin{tabular}{rll}
173 & [\verb|\dotfill|] & [\verb|\zdotfill|]\!
174 \dfctest{0.43em}
175 \dfctest{0.44em}
176 \dfctest{0.45em}
177 \dfctest{0.87em}
178 \dfctest{0.88em}
179 \dfctest{0.89em}
180 \dfctest{1.31em}
181 \dfctest{1.32em}
182 \dfctest{1.33em}
183 \end{tabular}
184 \ChapterStop
185 \end{document}
186 </example>

```

## 6 Implementation

### 6.1 Package zref

#### 6.1.1 Identification

```

187 <*package>
188 \NeedsTeXFormat{LaTeX2e}
189 \ProvidesPackage{zref}
190 [2011/03/18 v2.21 New reference scheme for LaTeX2e (HO)]%

```

### 6.1.2 Load basic module

```
191 \RequirePackage{zref-base}[2011/03/18]
```

Abort package loading if zref-base could not be loaded successfully.

```
192 \@ifundefined{ZREF@base@ok}{\endinput}{}

```

### 6.1.3 Process options

Known modules are loaded and the release date is checked.

```

193 \def\ZREF@temp#1{%
194   \DeclareOption{#1}{%
195     \AtEndOfPackage{%
196       \RequirePackage{zref-#1}[2011/03/18]%
197     }%
198   }%
199 }
200 \ZREF@temp{abspage}
201 \ZREF@temp{counter}
202 \ZREF@temp{dotfill}
203 \ZREF@temp{hyperref}
204 \ZREF@temp{lastpage}
205 \ZREF@temp{marks}
206 \ZREF@temp{nextpage}
207 \ZREF@temp{pageattr}
208 \ZREF@temp{pagelayout}
209 \ZREF@temp{perpage}
210 \ZREF@temp{runs}
211 \ZREF@temp{savepos}
212 \ZREF@temp{thepage}
213 \ZREF@temp{titleref}
214 \ZREF@temp{totpages}
215 \ZREF@temp{user}
216 \ZREF@temp{xr}

217 \ProcessOptions\relax
218 </package>

```

## 6.2 Module base

### 6.2.1 Prefixes

This package uses the following prefixes for macro names:

**\zref@:** Macros of the programmer's interface.

**\ZREF@:** Internal macros.

**\Z@L@listname:** The properties of the list *<listname>*.

**\Z@D@propname:** The default value for property *<propname>*.

**\Z@E@propname:** Extract function for property *<propname>*.

**\Z@X@propname:** Information whether a property value for property *<propname>* is expanded immediately or at shipout time.

**\Z@C@propname:** Current value of the property *<propname>*.

**\Z@R@labelname:** Data for reference *<labelname>*.

**\ZREF@org@:** Original versions of patched commands.

`\z`: For macros in user land, defined if module `user` is set.

The following family names are used for keys defined according to the keyval package:

`ZREF@TR`: Setup for module `titleref`.

## 6.2.2 Identification

```
219 (*base)
220 \NeedsTeXFormat{LaTeX2e}
221 \ProvidesPackage{zref-base}%
222 [2011/03/18 v2.21 Module base for zref (H0)]%
```

## 6.2.3 Utilities

```
223 \RequirePackage{ltxcmds}[2010/03/01]
224 \RequirePackage{infwarerr}[2010/04/08]
225 \RequirePackage{kvsetkeys}[2010/03/01]
226 \RequirePackage{kvdefinekeys}[2010/03/01]
227 \RequirePackage{pdftexcmds}[2010/04/01]
```

`\ZREF@name` Several times the package name is used, thus we store it in `\ZREF@name`.

```
228 \def\ZREF@name{zref}

229 \ltx@ifundefined{protected}{%
230   \RequirePackage{makerobust}[2006/03/18]%
```

`\ZREF@Robust`

```
231   \def\ZREF@Robust#1#2{%
232     \def\ZREF@temp{\MakeRobustcommand#2}%
233     \afterassignment\ZREF@temp
234     #1#2%
235   }%

236 }%
```

`\ZREF@Robust`

```
237   \def\ZREF@Robust#1{%
238     \protected#1%
239   }%

240 }
```

`\ZREF@ifDefinable`

```
241 \def\ZREF@ifDefinable#1#2#3{%
242   \@ifdefinable{#1}{%
243     \ZREF@Robust{#2}#1#3%
244   }%
245 }
```

`\ZREF@UpdatePdfTeX` `\ZREF@UpdatePdfTeX` is used as help message text in error messages.

```
246 \def\ZREF@UpdatePdfTeX{Update pdfTeX.}
```

`\ifZREF@found` The following switch is used in list processing.

```
247 \newif\ifZREF@found
```

`\ZREF@patch` Macro `\ZREF@patch` first checks the existence of the command and safes it.

```
248 \def\ZREF@patch#1{%
249   \ltx@ifundefined{#1}{%
250     \ltx@gobble
251   }{%
252     \expandafter\let\csname ZREF@org@#1\expandafter\endcsname
253     \csname #1\endcsname
254     \ltx@firstofone
255   }%
256 }
```

## 6.2.4 Check for $\varepsilon$ -TeX

The use of  $\varepsilon$ -TeX should be standard nowadays for L<sup>A</sup>T<sub>E</sub>X. We test for  $\varepsilon$ -TeX in order to use its features later.

```
257 \ltx@ifundefined{eTeXversion}{%
258   \PackageError{ZREF@name}{%
259     Missing support for eTeX; package is abandoned%
260   }{%
261     Use a TeX compiler that support eTeX and enable eTeX %
262     in the format.%
263   }%
264   \endinput
265 }{}%

266 \RequirePackage{etexcmds}[2007/09/09]
267 \ifetex@unexpanded
268 \else
269   \PackageError{ZREF@name}{%
270     Missing e-TeX's \string\unexpanded.\MessageBreak
271     Add \string\RequirePackage\string{etexcmds\string} before %
272     \string\documentclass%
273   }{%
274     Probably you are using some package (e.g. ConTeXt) that %
275     redefines \string\unexpanded%
276   }%
277   \expandafter\endinput
278 \fi
```

## 6.2.5 Auxiliary file stuff

We are using some commands in the .aux files. However sometimes these auxiliary files are interpreted by L<sup>A</sup>T<sub>E</sub>X processes that haven't loaded this package (e.g. package xr). Therefore we provide dummy definitions.

```
279 \RequirePackage{auxhook}
280 \AddLineBeginAux{%
281   \string\providecommand\string\zref@newlabel[2]{}%
282 }
```

`\ZREF@RefPrefix`

```
283 \def\ZREF@RefPrefix{Z@R}
```

`\zref@newlabel` For the implementation of `\zref@newlabel` we call the same internal macro `\@newl@bel` that is used in `\newlabel`. Thus we have for free:

- `\Z@R@labelname` is defined.
- L<sup>A</sup>T<sub>E</sub>X's check for multiple references.
- L<sup>A</sup>T<sub>E</sub>X's check for changed references.

```
284 \ZREF@Robust\edef\zref@newlabel{%
285   \noexpand\@newl@bel{\ZREF@RefPrefix}%
286 }
```

## 6.2.6 Property lists

`\zref@newlist` Property lists are stored as list of property names enclosed in curly braces. `\zref@newlist` creates a new list as empty list. Assignments to property lists are global.

```
287 \ZREF@Robust\def\zref@newlist#1{%
288   \zref@iflistundefined{#1}{%
289     \@ifdefinable{Z@L#1}{%
290       \global\expandafter\let\csname Z@L#1\endcsname\ltx@empty
```



```

291     \PackageInfo\ZREF@name{New property list: #1}%
292   }%
293 }{%
294   \PackageError\ZREF@name{%
295     Property list ‘#1’ already exists%
296   }\@ehc
297 }%
298 }

```

`\zref@iflistundefined` `\zref@iflistundefined` checks the existence of the property list #1. If the property list is present, then #2 is executed and #3 otherwise.

```

299 \def\zref@iflistundefined#1{%
300   \ltx@ifundefined{Z@L@#1}%
301 }

```

`\zref@listexists` `\zref@listexists` only executes #2 if the property list #1 exists and raises an error message otherwise.

```

302 \ZREF@Robust\def\zref@listexists#1{%
303   \zref@iflistundefined{#1}{%
304     \PackageError\ZREF@name{%
305       Property list ‘#1’ does not exist%
306     }\@ehc
307   }%
308 }

```

`\zref@iflistcontainsprop` `\zref@iflistcontainsprop` checks, whether a property #2 is already present in a property list #1.

```

309 \ZREF@Robust\def\zref@iflistcontainsprop#1#2{%
310   \zref@iflistundefined{#1}{%
311     \ltx@secondoftwo
312   }{%
313     \begingroup\expandafter\endgroup
314     \expandafter\in@
315     \csname#2\expandafter\expandafter\expandafter\endcsname
316     \expandafter\expandafter\expandafter{\csname Z@L@#1\endcsname}%
317     \csname ltx@ifin@ first\else second\fi oftwo\endcsname
318   }%
319 }

```

`\zref@listforloop`

```

320 \def\zref@listforloop#1#2{%
321   \zref@listexists{#1}{%
322     \expandafter\expandafter\expandafter\@tfor
323     \expandafter\expandafter\expandafter\zref@prop
324     \expandafter\expandafter\expandafter:%
325     \expandafter\expandafter\expandafter=%
326     \csname Z@L@#1\endcsname
327     \do{%
328       \begingroup
329       \escapechar=-1 %
330       \edef\x{\endgroup
331         \def\noexpand\zref@prop{%
332           \expandafter\string\zref@prop
333         }%
334       }%
335       \x
336       #2\zref@prop
337     }%
338   }%
339 }

```

`\zref@addprops` `\zref@addprop` adds the properties #2 to the property list #1, if the property is not already in the list. Otherwise a warning is given.

```

340 \ZREF@Robust\def\zref@addprops#1#2{%
341   \zref@listexists{#1}{%
342     \comma@parse{#2}{%
343       \zref@propexists\comma@entry{%
344         \zref@iflistcontainsprop{#1}\comma@entry{%
345           \PackageWarning\ZREF@name{%
346             Property '\comma@entry' is already in list '#1'%
347           }%
348         }%
349       \begingroup\expandafter\endgroup
350       \expandafter\g@addto@macro
351       \csname Z@L@#1\expandafter\endcsname
352       \expandafter{\csname\comma@entry\endcsname}%
353     }%
354   }%
355   \ltx@gobble
356 }%
357 }%
358 }

```

`\zref@addprop` `\zref@addprop` adds the property #2 to the property list #1, if the property is not already in the list. Otherwise a warning is given.

```

359 \ZREF@Robust\def\zref@addprop#1#2{%
360   \zref@listexists{#1}{%
361     \zref@propexists{#2}{%
362       \zref@iflistcontainsprop{#1}{#2}{%
363         \PackageWarning\ZREF@name{%
364           Property '#2' is already in list '#1'%
365         }%
366       }%
367     \begingroup\expandafter\endgroup
368     \expandafter\g@addto@macro
369     \csname Z@L@#1\expandafter\endcsname
370     \expandafter{\csname#2\endcsname}%
371   }%
372 }%
373 }%
374 }

```

`\zref@localaddprops`

```

375 \ZREF@Robust\def\zref@localaddprops#1#2{%
376   \zref@listexists{#1}{%
377     \comma@parse{#2}{%
378       \zref@propexists\comma@entry{%
379         \zref@iflistcontainsprop{#1}\comma@entry{%
380           \PackageWarning\ZREF@name{%
381             Property '\comma@entry' is already in list '#1'%
382           }%
383         }%
384       \begingroup\expandafter\endgroup
385       \expandafter\ltx@LocalAppendToMacro
386       \csname Z@L@#1\expandafter\endcsname
387       \expandafter{\csname\comma@entry\endcsname}%
388     }%
389   }%
390   \ltx@gobble
391 }%
392 }%
393 }

```

\zref@localaddprop

```
394 \ZREF@Robust\def\zref@localaddprop#1#2{%
395   \zref@listexists{#1}{%
396     \zref@propexists{#2}{%
397       \zref@iflistcontainsprop{#1}{#2}{%
398         \PackageWarning\ZREF@name{%
399           Property ‘#2’ is already in list ‘#1’%
400         }%
401       }%
402     \begingroup\expandafter\endgroup
403     \expandafter\ltx@LocalAppendToMacro
404     \csname Z@L@#1\expandafter\endcsname
405     \expandafter{\csname#2\endcsname}%
406   }%
407 }%
408 }%
409 }

410 \ltx@ifundefined{pdf@strcmp}{%
```

\zref@delprop

```
411 \ZREF@Robust\def\zref@delprop{%
412   \ZREF@delprop\gdef
413 }%
```

\zref@localdelprop

```
414 \ZREF@Robust\def\zref@localdelprop{%
415   \ZREF@delprop\def
416 }%
```

\ZREF@delprop

```
417 \def\ZREF@delprop#1#2#3{%
418   \zref@listexists{#2}{%
419     \begingroup
420     \escapechar=-1 %
421     \def\ZREF@param{#3}%
422     \@onelevel@sanitize\ZREF@param
423     \toks@{%
424       \expandafter\expandafter\expandafter\ZREF@@delprop
425       \csname Z@L@#2\endcsname!%
426     \expandafter\endgroup
427     \expandafter#1\csname Z@L@#2\expandafter\endcsname
428     \expandafter{%
429       \the\toks@
430     }%
431   }%
432 }%
```

\ZREF@@delprop

```
433 \def\ZREF@@delprop#1{%
434   \expandafter\ZREF@@delprop\expandafter{\string#1}#1%
435 }%
```

\ZREF@@@delprop

```
436 \def\ZREF@@@delprop#1#2{%
437   \ifx#2!%
438   \else
439     \def\ZREF@temp{#1}%
440     \@onelevel@sanitize\ZREF@temp
441     \ifx\ZREF@param\ZREF@temp
442     \else
```

```

443     \toks\expandafter{%
444     \the\expandafter\toks\csname#1\endcsname
445     }%
446     \fi
447     \expandafter\ZREF@delprop
448     \fi
449 }%

450 }{%

\zref@delprop
451 \ZREF@Robust\def\zref@delprop{%
452   \ZREF@delprop\xdef
453 }%

\zref@localdelprop
454 \ZREF@Robust\def\zref@localdelprop{%
455   \ZREF@delprop\edef
456 }%

\ZREF@delprop
457 \def\ZREF@delprop#1#2#3{%
458   \zref@listexists{#2}{%
459     \def\ZREF@param{#3}%
460     \edef\ZREF@SavedEscapechar{\the\escapechar}%
461     \escapechar=-1 %
462     \expandafter#1\csname Z@L@#2%
463     \expandafter\expandafter\expandafter\endcsname{%
464       \expandafter\expandafter\expandafter\ZREF@delprop
465       \csname Z@L@#2\endcsname!%
466     }%
467     \escapechar=\ZREF@SavedEscapechar\relax
468   }%
469 }%

```

`\ZREF@@delprop` Caution: #1 might be an `\if` or similar token.

```

470 \def\ZREF@@delprop#1{%
471   \expandafter\ZREF@@delprop\expandafter{\string#1}#1%
472 }%

```

`\ZREF@@@delprop`

```

473 \def\ZREF@@@delprop#1#2{%
474   \ifx#2!%
475   \else
476     \ifnum\pdf@strcmp{#1}{\ZREF@param}=\ltx@zero
477     \else
478       \expandafter\noexpand\csname#1\endcsname
479     \fi
480     \expandafter\ZREF@@delprop
481   \fi
482 }%

483 }

```

### 6.2.7 Properties

`\zref@ifpropundefined` `\zref@ifpropundefined` checks the existence of the property #1. If the property is present, then #2 is executed and #3 otherwise.

```

484 \def\zref@ifpropundefined#1{%
485   \ltx@ifundefined{Z@E@#1}%
486 }

```

`\zref@propexists` Some macros rely on the existence of a property. `\zref@propexists` only executes #2 if the property #1 exists and raises an error message otherwise.

```

487 \ZREF@Robust\def\zref@propexists#1{%
488   \zref@ifpropundefined{#1}{%
489     \PackageError\ZREF@name{%
490       Property ‘#1’ does not exist%
491     }\@ehc
492   }%
493 }

```

`\zref@newprop` A new property is declared by `\zref@newprop`, the property name  $\langle propname \rangle$  is given in #1. The property is created and configured. If the star form is given, then the expansion of the property value is delayed to page shipout time, when the reference is written to the .aux file.

`\Z@D@propname`: Stores the default value for this property.

`\Z@E@propname`: Extract function.

`\Z@X@propname`: Information whether the expansion of the property value is delayed to shipout time.

`\Z@C@propname`: Current value of the property.

```

494 \ZREF@Robust\def\zref@newprop{%
495   \@ifstar{%
496     \let\ZREF@X\noexpand
497     \ZREF@newprop
498   }{%
499     \let\ZREF@X\ltx@empty
500     \ZREF@newprop
501   }%
502 }

```

`\ZREF@newprop`

```

503 \def\ZREF@newprop#1{%
504   \edef\ZREF@P{#1}%
505   \@onelevel@sanitize\ZREF@P
506   \begingroup
507   \ifx\ZREF@P\ZREF@par
508     \PackageError\ZREF@name{%
509       Invalid property name ‘\ZREF@P’%
510     }{%
511       The property name ‘par’ is not allowed %
512       because of internal reasons.%
513       \MessageBreak
514       \@ehc
515     }%
516     \def\ZREF@@newprop[##1]##2{\endgroup}%
517   \else
518     \zref@ifpropundefined\ZREF@P{%
519       \endgroup
520       \PackageInfo\ZREF@name{%
521         New property: \ZREF@P
522       }%
523     }{%
524       \PackageError\ZREF@name{%
525         Property ‘\ZREF@P’ already exists%
526       }\@ehc
527       \def\ZREF@@newprop[##1]##2{\endgroup}%
528     }%
529   \fi
530   \@ifnextchar[\ZREF@@newprop{\ZREF@@newprop[\zref@default]}%
531 }

```

`\ZREF@par`

```
532 \def\ZREF@par{par}
533 \@onelevel@sanitize\ZREF@par
```

`\ZREF@@newprop`

```
534 \def\ZREF@@newprop[#1]{%
535   \global\@namedef{Z@D@\ZREF@P}{#1}%
536   \global\expandafter\let\csname Z@X@\ZREF@P\endcsname\ZREF@X
537   \begingroup\expandafter\endgroup
538   \expandafter\ZREF@@@newprop\csname\ZREF@P\endcsname
539   \expandafter\gdef\csname Z@C@\ZREF@P\endcsname{}%
540   \zref@setcurrent\ZREF@P
541 }
542 \def\ZREF@@@newprop#1{%
543   \expandafter
544   \gdef\csname Z@E@\ZREF@P\endcsname##1#1##2##3\ZREF@nil{##2}%
545 }
```

`\zref@setcurrent` `\zref@setcurrent` sets the current value for a property.

```
546 \ZREF@Robust\def\zref@setcurrent#1#2{%
547   \zref@propexists{#1}{%
548     \expandafter\def\csname Z@C@#1\endcsname{#2}%
549   }%
550 }
```

`\ZREF@getcurrent` `\zref@getcurrent` gets the current value for a property.

```
551 \def\ZREF@getcurrent#1{%
552   \romannumeral0%
553   \ltx@ifundefined{Z@C@#1}{%
554     \ltx@space
555   }{%
556     \expandafter\expandafter\expandafter\ltx@space
557     \csname Z@C@#1\endcsname
558   }%
559 }
```

`\ZREF@u@getcurrent`

```
560 \def\ZREF@wu@getcurrent#1{%
561   \etex@unexpanded\expandafter\expandafter\expandafter{%
562     \ZREF@getcurrent{#1}%
563   }%
564 }
```

`\zref@getcurrent`

```
565 \let\zref@getcurrent\ZREF@getcurrent
```

### 6.2.8 Reference generation

`\zref@label` Label macro that uses the main property list.

```
566 \ZREF@Robust\def\zref@label#1{%
567   \zref@labelbylist{#1}\ZREF@mainlist
568 }
```

`\zref@labelbylist` Label macro that stores the properties, specified in the property list #2.

```
569 \ZREF@Robust\def\zref@labelbylist#1#2{%
570   \@bsphack
571   \zref@listexists{#2}{%
572     \expandafter\expandafter\expandafter\ZREF@label
573     \expandafter\expandafter\expandafter{%
574       \csname Z@L@#2\endcsname
```

```

575     }{#1}%
576   }%
577   \@esphack
578 }

```

`\zref@labelbyprops` The properties are directly specified in a comma separated list.

```

579 \ZREF@Robust\def\zref@labelbyprops#1#2{%
580   \@bsphack
581   \begingroup
582     \toks@{%
583       \comma@parse{#2}{%
584         \zref@ifpropundefined\comma@entry{%
585           \PackageWarning\ZREF@name{%
586             Property '\comma@entry' is not known%
587           }%
588         }{%
589           \toks@\expandafter{%
590             \the\expandafter\toks@\csname\comma@entry\endcsname
591           }%
592         }%
593       \ltx@gobble
594     }%
595   \expandafter\endgroup
596   \expandafter\ZREF@label\expandafter{\the\toks@}{#1}%
597   \@esphack
598 }

```

`\zref@labelbykv`

```

599 \ZREF@Robust\def\zref@labelbykv#1#2{%
600   \@bsphack
601   \begingroup
602     \let\Z@L@ZREF@temp\ltx@empty
603     \kvsetkeys{ZREF@LABEL}{#1}%
604     \ifZREF@immediate
605       \expandafter\zref@wrapper@immediate\expandafter{%
606         \expandafter\ZREF@label\expandafter{\Z@L@ZREF@temp}{#2}%
607       }%
608     \else
609       \expandafter\ZREF@label\expandafter{\Z@L@ZREF@temp}{#2}%
610     \fi
611   \endgroup
612   \@esphack
613 }

614 \kv@define@key{ZREF@LABEL}{prop}{%
615   \edef\ZREF@param{#1}%
616   \zref@propexists\ZREF@param{%
617     \zref@iflistcontainsprop{ZREF@temp}\ZREF@param}{%
618       \begingroup\expandafter\endgroup
619       \expandafter\ltx@LocalAppendToMacro
620       \expandafter\Z@L@ZREF@temp
621       \expandafter{\csname\ZREF@param\endcsname}%
622     }%
623   }%
624 }

625 \kv@define@key{ZREF@LABEL}{list}{%
626   \zref@listforloop{#1}{%
627     \zref@iflistcontainsprop{ZREF@temp}\zref@prop}{%
628       \begingroup\expandafter\endgroup
629       \expandafter\ltx@LocalAppendToMacro
630       \expandafter\Z@L@ZREF@temp
631       \expandafter{\csname\zref@prop\endcsname}%

```

```

632   }%
633   \ltx@gobble
634 }%
635 }
636 \kv@define@key{ZREF@LABEL}{delprop}{%
637   \zref@propexists{#1}{%
638     \zref@localdelprop{ZREF@temp}{#1}%
639   }%
640 }
641 \kv@define@key{ZREF@LABEL}{immediate}[true]{%
642   \edef\ZREF@param{#1}%
643   \ifx\ZREF@param\ZREF@true
644     \ZREF@immediatetrue
645   \else
646     \ifx\ZREF@param\ZREF@false
647       \ZREF@immediatefalse
648     \else
649       \PackageWarning\ZREF@name{%
650         Option 'immediate' expects 'true' or 'false'.\MessageBreak
651         Ignoring invalid value '\ZREF@param'%
652       }%
653     \fi
654   \fi
655 }

```

`\ZREF@false`

```
656 \def\ZREF@false{false}
```

`\ZREF@true`

```
657 \def\ZREF@true{true}
```

```

658 \kv@define@key{ZREF@LABEL}{values}[]{%
659   \kv@parse{#1}{%
660     \ifx\kv@value\relax
661       \@PackageWarning\ZREF@name{%
662         Missing value for property '\kv@key'%
663       }%
664     \expandafter\ltx@gobbletwo
665   \else
666     \expandafter\zref@setcurrent
667   \fi
668 }%
669 }

```

`\ifZREF@immediate` The switch `\ifZREF@immediate` tells us, whether the label should be written immediately or at page shipout time. `\ZREF@label` need to be notified about this, because it must disable the deferred execution of property values, if the label is written immediately.

```
670 \newif\ifZREF@immediate
```

`\zref@wrapper@immediate` The argument of `\zref@wrapper@immediate` is executed inside a group where `\write` is redefined by adding `\immediate` before its execution. Also `\ZREF@label` is notified via the switch `\ifZREF@immediate`.

```

671 \ZREF@Robust{\long\def}\zref@wrapper@immediate#1{%
672   \begingroup
673     \ZREF@immediatetrue
674     \let\ZREF@org@write\write
675     \def\write{\immediate\ZREF@org@write}%
676     #1%
677   \endgroup
678 }

```



`\ZREF@label` `\ZREF@label` writes the data in the `.aux` file. #1 contains the list of valid properties, #2 the name of the reference. In case of immediate writing, the deferred execution of property values is disabled. Also `32is` is made expandable in this case.

```

679 \def\ZREF@label#1#2{%
680   \if@filesw
681     \begingroup
682       \ifZREF@immediate
683         \let\ZREF@org@thepage\thepage
684       \fi
685     \protected@write\@auxout{%
686       \ifZREF@immediate
687         \let\thepage\ZREF@org@thepage
688       \fi
689       \let\ZREF@temp\ltx@empty
690       \@tfor\ZREF@P:=#1\do{%
691         \begingroup
692           \escapechar=-1 %
693           \edef\x{\endgroup
694             \def\noexpand\ZREF@P{%
695               \expandafter\string\ZREF@P
696             }%
697           }%
698         \x
699         \expandafter\ifx
700           \csname
701             \ifZREF@immediate
702               relax%
703             \else
704               Z@X@\ZREF@P%
705             \fi
706           \endcsname
707           \noexpand
708           \expandafter\let\csname Z@C@\ZREF@P\endcsname\relax
709         \fi
710         \toks@\expandafter{\ZREF@temp}%
711         \edef\ZREF@temp{%
712           \the\toks@
713           \ltx@backslashchar\ZREF@P{%
714             \expandafter\noexpand\csname Z@C@\ZREF@P\endcsname
715           }%
716         }%
717       }%
718     }-%
719     \string\zref@newlabel{#2}{\ZREF@temp}%
720   }%
721 \endgroup
722 \fi
723 }
724 \def\ZREF@addtoks#1{%
725   \toks@\expandafter\expandafter\expandafter{%
726     \expandafter\the\expandafter\toks@#1%
727   }%
728 }

```

## 6.2.9 Reference querying and extracting

Design goal for the extracting macros is that the extraction process is full expandable. Thus these macros can be used in expandable contexts. But there are problems that cannot be solved by full expandable macros:

- In standard L<sup>A</sup>T<sub>E</sub>X undefined references sets a flag and generate a warning. Both actions are not expandable.

- Babel's support for its shorthand uses commands that use non-expandable assignments. However currently there is hope, that primitives are added to pdfTeX that allows the detection of contexts. Then the shorthand can detect, if they are executed inside \csname and protect themselves automatically.

`\zref@ifrefundefined` If a reference #1 is undefined, then macro `\zref@ifrefundefined` calls #2 and #3 otherwise.

```
729 \def\zref@ifrefundefined#1{%
730   \ltx@ifundefined{Z@R@#1}%
731 }
```

`\zifrefundefined` If a reference #1 is undefined, then macro `\zref@ifrefundefined` calls #2 and #3 otherwise. Also the reference is marked used.

```
732 \ZREF@ifDefinable\zifrefundefined\def{%
733   #1{%
734     \zref@wrapper@babel\ZREF@ifrefundefined{#1}%
735   }%
736 }
```

`\ZREF@ifrefundefined`

```
737 \def\ZREF@ifrefundefined#1{%
738   \zref@refused{#1}%
739   \zref@ifrefundefined{#1}%
740 }
```

`\zref@refused` The problem with undefined references is addressed by the macro `\zref@refused`. This can be used outside the expandable context. In case of an undefined reference the flag is set to notify L<sup>A</sup>T<sub>E</sub>X and a warning is given.

```
741 \ZREF@Robust\def\zref@refused#1{%
742   \zref@wrapper@babel\ZREF@refused{#1}%
743 }
```

`\ZREF@refused`

```
744 \def\ZREF@refused#1{%
745   \zref@ifrefundefined{#1}{%
746     \protect\G@refundefinedtrue
747     \@latex@warning{%
748       Reference ‘#1’ on page \thepage \space undefined%
749     }%
750   }{}%
751 }
```

`\zref@ifrefcontainsprop` `\zref@ifrefcontainsprop` looks, if the reference #1 has the property #2 and calls then #3 and #4 otherwise.

```
752 \def\zref@ifrefcontainsprop#1#2{%
753   \zref@ifrefundefined{#1}{%
754     \ltx@secondoftwo
755   }{%
756     \expandafter\ZREF@ifrefcontainsprop
757     \csname Z@E@#2\expandafter\endcsname
758     \csname#2\expandafter\expandafter\expandafter\endcsname
759     \expandafter\expandafter\expandafter{%
760       \csname Z@R@#1\endcsname
761     }%
762   }%
763 }
764 \def\ZREF@ifrefcontainsprop#1#2#3{%
765   \expandafter\ifx\expandafter\ZREF@novalue
766   #1#3#2\ZREF@novalue\ZREF@nil\ltx@empty
767   \expandafter\ltx@secondoftwo
```

```

768 \else
769   \expandafter\ltx@firstoftwo
770 \fi
771 }
772 \def\ZREF@novalue{\ZREF@NOVALUE}

```

`\zref@extract` `\zref@extract` is an abbreviation for the case that the default of the property is used as default value.

```

773 \def\ZREF@extract#1#2{%
774   \romannumeral0%
775   \ltx@ifundefined{Z@D@#2}{%
776     \expandafter\ltx@space\zref@default
777   }{%
778     \expandafter\expandafter\expandafter\ZREF@@extract
779     \expandafter\expandafter\expandafter{%
780       \csname Z@D@#2\endcsname
781     }{#1}{#2}%
782   }%
783 }

```

`\ZREF@@extract`

```

784 \def\ZREF@@extract#1#2#3{%
785   \expandafter\expandafter\expandafter\ltx@space
786   \zref@extractdefault{#2}{#3}{#1}%
787 }

```

`\ZREF@wu@extract`

```

788 \def\ZREF@wu@extract#1#2{%
789   \etex@unexpanded\expandafter\expandafter\expandafter{%
790     \ZREF@extract{#1}{#2}%
791   }%
792 }

```

`\zref@extract`

```

793 \let\zref@extract\ZREF@extract

```

`\ZREF@extractdefault` The basic extracting macro is `\zref@extractdefault` with the reference name in #1, the property in #2 and the default value in #3 in case for problems.

```

794 \def\ZREF@extractdefault#1#2#3{%
795   \romannumeral0%
796   \zref@ifrefundefined{#1}\ltx@firstoftwo{%
797     \zref@ifpropundefined{#2}\ltx@firstoftwo\ltx@secondoftwo
798   }{%
799     \ltx@space
800     #3%
801   }{%
802     \expandafter\expandafter\expandafter\ltx@space
803     \csname Z@E@#2\expandafter\expandafter\expandafter\endcsname
804     \csname Z@R@#1\expandafter\endcsname
805     \csname#2\endcsname{#3}\ZREF@nil
806   }%
807 }

```

`\ZREF@wu@extractdefault`

```

808 \def\ZREF@wu@extractdefault#1#2#3{%
809   \etex@unexpanded\expandafter\expandafter\expandafter{%
810     \ZREF@extractdefault{#1}{#2}{#3}%
811   }%
812 }

```

`\zref@extractdefault`

```

813 \let\zref@extractdefault\ZREF@extractdefault

```

`\ZREF@wrapper@unexpanded`

```
814 \ZREF@Robust{\long\def}\ZREF@wrapper@unexpanded#1{%
815   \let\zref@wrapper@unexpanded\ltx@firstofone
816   \let\zref@getcurrent\ZREF@wu@getcurrent
817   \let\zref@extractdefault\ZREF@wu@extractdefault
818   \let\zref@extract\ZREF@wu@extract
819   #1%
820   \let\zref@wrapper@unexpanded\ZREF@wrapper@unexpanded
821   \let\zref@getcurrent\ZREF@getcurrent
822   \let\zref@extractdefault\ZREF@extractdefault
823   \let\zref@extract\ZREF@extract
824 }
```

`\zref@wrapper@unexpanded`

```
825 \ltx@ifundefined{etex@unexpanded}{%
826   \let\zref@wrapper@unexpanded\ltx@firstofone
827 }{%
828   \let\zref@wrapper@unexpanded\ZREF@wrapper@unexpanded
829 }
```

### 6.2.10 Compatibility with babel

`\zref@wrapper@babel`

```
830 \ZREF@Robust{\long\def}\zref@wrapper@babel#1#2{%
831   \ifcsname if@safe@actives\endcsname
832     \expandafter\ltx@firstofone
833   \else
834     \expandafter\ltx@secondoftwo
835   \fi
836   {%
837     \if@safe@actives
838       \expandafter\ltx@secondoftwo
839     \else
840       \expandafter\ltx@firstoftwo
841     \fi
842     {%
843       \begingroup
844         \csname @safe@activestrue\endcsname
845         \edef\x{#2}%
846         \expandafter\endgroup
847         \expandafter\ZREF@wrapper@babel\expandafter{\x}{#1}%
848       }%
849     }{%
850       #1{#2}%
851     }%
852 }
853 \long\def\ZREF@wrapper@babel#1#2{%
854   #2{#1}%
855 }
```

### 6.2.11 Unique counter support

`\zref@require@unique` Generate the counter `zref@unique` if the counter does not already exist.

```
856 \ZREF@Robust\def\zref@require@unique{%
857   \@ifundefined{c@zref@unique}{%
858     \begingroup
859     \let\@addtoreset\ltx@gobbletwo
860     \newcounter{zref@unique}%
861     \endgroup
```

`\thezref@unique` `\thezref@unique` is used for automatically generated unique labelnames.

```

862   \renewcommand*{\thezref@unique}{%
863     zref@number\c@zref@unique
864   }%
865   }{}%
866 }

```

### 6.2.12 Utilities

`\ZREF@number`

```

867 \ltx@ifundefined{numexpr}{%
868   \let\ZREF@number\number
869 }{}%
870 \def\ZREF@number#1{\the\numexpr#1}%
871 }

```

### 6.2.13 Setup

`\zref@setdefault` Standard L<sup>A</sup>T<sub>E</sub>X prints “??” in bold face if a reference is not known. `\zref@default` holds the text that is printed in case of unknown references and is used, if the default was not specified during the definition of the new property by `\ref@newprop`. The global default value can be set by `\zref@setdefault`.

```

872 \ZREF@Robust\def\zref@setdefault#1{%
873   \def\zref@default{#1}%
874 }

```

`\zref@default` Now we initialize `\zref@default` with the same value that L<sup>A</sup>T<sub>E</sub>X uses for its undefined references.

```

875 \zref@setdefault{%
876   \nfss@text{\reset@font\bfseries ??}%
877 }

```

#### Main property list.

`\zref@setmainlist` The name of the default property list is stored in `\ZREF@mainlist` and can be set by `\zref@setmainlist`.

```

878 \ZREF@Robust\def\zref@setmainlist#1{%
879   \def\ZREF@mainlist{#1}%
880 }
881 \zref@setmainlist{main}

```

Now we create the list.

```

882 \zref@newlist\ZREF@mainlist

```

**Main properties.** The two properties `default` and `page` are created and added to the main property list. They store the data that standard L<sup>A</sup>T<sub>E</sub>X uses in its references created by `\label`.

`default` the appearance of the latest counter that is incremented by `\refstepcounter`

`page` the appearance of the page counter

```

883 \zref@newprop{default}{\@currentlabel}
884 \zref@newprop*{page}{\thepage}
885 \zref@addprops\ZREF@mainlist{default,page}

```

## Properties

`\ZREF@NewPropAnchor`

```
886 \def\ZREF@NewPropAnchor{%
887   \zref@newprop{anchor}{%
888     \ltx@ifundefined{@currentHref}{}\@currentHref}%
889   }%
890 \global\let\ZREF@NewPropAnchor\relax
891 }
```

`\zref@titleref@current` Later we will redefine the section and caption macros to catch the current title and remember the value in `\zref@titleref@current`.

`\ZREF@NewPropTitle`

```
892 \def\ZREF@NewPropTitle{%
893   \gdef\zref@titleref@current{}%
894   \zref@newprop{title}{\zref@titleref@current}%
895   \global\let\ZREF@NewPropTitle\relax
896 }
```

`\ZREF@NewPropTheotype`

```
897 \def\ZREF@NewPropTheotype{%
898   \zref@newprop{theotype}{}%
899   \global\let\ZREF@NewPropTheotype\relax
900 }
```

## Mark successful loading

```
901 \let\ZREF@base@ok=Y
902 \</base>
```

## 6.3 Module user

```
903 <*user>
904 \NeedsTeXFormat{LaTeX2e}
905 \ProvidesPackage{zref-user}%
906 [2011/03/18 v2.21 Module user for zref (HO)]%
907 \RequirePackage{zref-base}[2011/03/18]
908 \ifx\ZREF@base@ok Y%
909 \else
910 \expandafter\endinput
911 \fi
```

Module user enables a small user interface. All macros are prefixed by `\z`.

First we define the pendants to the standard L<sup>A</sup>T<sub>E</sub>X referencing commands `\label`, `\ref`, and `\pageref`.

`\zlabel` Similar to `\label` the macro `\zlabel` writes a reference entry in the `.aux` file. The main property list is used. Also we add the babel patch. The `\label` command can also be used inside section titles, but it must not go into the table of contents. Therefore we have to check this situation.

```
912 \newcommand*\zlabel{%
913   \ifx\label\ltx@gobble
914     \expandafter\ltx@gobble
915   \else
916     \expandafter\zref@wrapper@babel\expandafter\zref@label
917   \fi
918 }%
```

`\zkvlabel`

```
919 \newcommand*\zkvlabel}[1]{%
920   \ifx\label\ltx@gobble
921     \expandafter\ltx@gobblethree
```

```

922 \fi
923 \zref@wrapper@babel{\zref@labelbykv{#1}}%
924 }%

```

`\zref` Macro `\zref` is the corresponding macro for `\ref`. Also it provides an optional argument in order to select another property.

```

925 \newcommand*{\zref}[2][default]{% robust because of optional argument
926 \zref@propexists{#1}{%
927 \zref@wrapper@babel\ZREF@zref{#2}{#1}}%
928 }%
929 }%
930 \def\ZREF@zref#1{%
931 \zref@refused{#1}%
932 \zref@extract{#1}%
933 }%

```

`\zpageref` For macro `\zpageref` we just call `\zref` with property `page`.

```

934 \ZREF@IfDefinable\zpageref\def{%
935 {\zref [page]}%
936 }

```

`\zrefused` For the following expandible user macros `\zrefused` should be used to notify L<sup>A</sup>T<sub>E</sub>X in case of undefined references.

```

937 \ZREF@IfDefinable\zrefused\def{%
938 {\zref@refused}%
939 }

940 </user>

```

## 6.4 Module `abspage`

```

941 <*abspage>
942 \NeedsTeXFormat{LaTeX2e}
943 \ProvidesPackage{zref-abspage}%
944 [2011/03/18 v2.21 Module abspage for zref (HO)]%
945 \RequirePackage{zref-base}[2011/03/18]
946 \ifx\ZREF@base@ok Y%
947 \else
948 \expandafter\endinput
949 \fi

```

Module `abspage` adds a new property `abspage` to the main property list for absolute page numbers. These are recorded by the help of package `atbegshi`.

```

950 \RequirePackage{atbegshi}%

```

The counter `abspage` must not go in the clear list of `@ckpt` that is used to set counters in `.aux` files of included T<sub>E</sub>X files.

```

951 \begingroup
952 \let\@addtoreset\ltx@gobbletwo
953 \newcounter{abspage}%
954 \endgroup
955 \setcounter{abspage}{0}%
956 \AtBeginShipout{%
957 \stepcounter{abspage}%
958 }%
959 \zref@newprop*{abspage}[0]{\the\c@abspage}%
960 \zref@addprop\ZREF@mainlist{abspage}%

```

Note that counter `abspage` shows the previous page during page processing. Before shipout the counter is incremented. Thus the property is correctly written with deferred writing. If the counter is written using `\zref@wrapper@immediate`, then the number is too small by one.

```

961 </abspage>

```

## 6.5 Module counter

```
962 <*counter>
963 \NeedsTeXFormat{LaTeX2e}
964 \ProvidesPackage{zref-counter}%
965 [2011/03/18 v2.21 Module counter for zref (HO)]%
966 \RequirePackage{zref-base}[2011/03/18]
967 \ifx\ZREF@base@ok Y%
968 \else
969 \expandafter\endinput
970 \fi
```

For features such as hyperref's `\autoref` we need the name of the counter. The property `counter` is defined and added to the main property list.

```
971 \zref@newprop{counter}{}
972 \zref@addprop\ZREF@mainlist{counter}
```

`\refstepcounter` is the central macro where we know which counter is responsible for the reference.

```
973 \AtBeginDocument{%
974 \ZREF@patch{refstepcounter}{%
975 \def\refstepcounter#1{%
976 \zref@setcurrent{counter}{#1}%
977 \ZREF@org@refstepcounter{#1}%
978 }%
979 }%
980 }
981 </counter>
```

## 6.6 Module lastpage

```
982 <*lastpage>
983 \NeedsTeXFormat{LaTeX2e}
984 \ProvidesPackage{zref-lastpage}%
985 [2011/03/18 v2.21 Module lastpage for zref (HO)]%
986 \RequirePackage{zref-base}[2011/03/18]
987 \RequirePackage{zref-abspage}[2011/03/18]
988 \RequirePackage{atveryend}[2009/12/07]
989 \ifx\ZREF@base@ok Y%
990 \else
991 \expandafter\endinput
992 \fi
```

The module `lastpage` implements the service of package `lastpage` by setting a reference `LastPage` at the end of the document. If module `abspage` is given, also the absolute page number is available, because the properties of the main property list are used.

```
993 \zref@newlist{LastPage}
994 \AfterLastShipout{%
995 \if@filesw
996 \begingroup
997 \advance\c@page\m@ne
998 \toks@\expandafter\expandafter\expandafter{%
999 \expandafter\Z@L@main
1000 \Z@L@LastPage
1001 }%
1002 \expandafter\zref@wrapper@immediate\expandafter{%
1003 \expandafter\ZREF@label\expandafter{\the\toks@}{LastPage}%
1004 }%
1005 \endgroup
1006 \fi
1007 }
```

```
\zref@iflastpage
```

```
1008 \def\zref@iflastpage#1{%
```



```

1009 \ifnum\zref@extractdefault{#1}{abspage}{-1}=%
1010     \zref@extractdefault{LastPage}{abspage}{-2} %
1011     \expandafter\ltx@firstoftwo
1012 \else
1013     \expandafter\ltx@secondoftwo
1014 \fi
1015 }

```

\ziflastpage

```

1016 \ZREF@ifDefinable\ziflastpage\def{%
1017   {\zref@wrapper@babel\ZREF@iflastpage}%
1018 }

```

ZREF@iflastpage

```

1019 \def\ZREF@iflastpage#1{%
1020   \zref@refused{LastPage}%
1021   \zref@refused{#1}%
1022   \zref@iflastpage{#1}%
1023 }

```

```

1024 </lastpage>

```

## 6.7 Module thepage

```

1025 <*thepage>
1026 \NeedsTeXFormat{LaTeX2e}
1027 \ProvidesPackage{zref-thepage}%
1028   [2011/03/18 v2.21 Module thepage for zref (HO)]%
1029 \RequirePackage{zref-base}[2011/03/18]
1030 \ifx\ZREF@base@ok Y%
1031 \else
1032   \expandafter\endinput
1033 \fi

1034 \RequirePackage{atbegshi}
1035 \RequirePackage{zref-abspage}[2011/03/18]

1036 \zref@newlist{thepage}
1037 \zref@addprop{thepage}{page}
1038 \AtBeginShipout{%
1039   \zref@wrapper@immediate{%
1040     \zref@labelbylist{thepage\the\value{abspage}}{thepage}%
1041   }%
1042 }

```

\zref@thepage@name

```

1043 \ltx@ifUndefined{numexpr}{%
1044   \def\zref@thepage@name#1{thepage\number#1}%
1045 }{%
1046   \def\zref@thepage@name#1{thepage\the\numexpr#1}%
1047 }

```

\zref@thepage

```

1048 \def\zref@thepage#1{%
1049   \zref@extract{\zref@thepage@name{#1}}{page}%
1050 }%

```

\zref@thepage@refused

```

1051 \ZREF@Robust\def\zref@thepage@refused#1{%
1052   \zref@refused{\zref@thepage@name{#1}}%
1053 }%

```

\zthepage

```
1054 \ZREF@ifDefinable\zthepage\def{%
1055   #1{%
1056     \zref@thepage@refused{#1}%
1057     \zref@thepage{#1}%
1058   }%
1059 }

1060 </thepage>
```

## 6.8 Module nextpage

```
1061 <*nextpage>
1062 \NeedsTeXFormat{LaTeX2e}
1063 \ProvidesPackage{zref-nextpage}%
1064 [2011/03/18 v2.21 Module nextpage for zref (HO)]%
1065 \RequirePackage{zref-base}[2011/03/18]
1066 \ifx\ZREF@base@ok Y%
1067 \else
1068   \expandafter\endinput
1069 \fi

1070 \RequirePackage{zref-abspage}[2011/03/18]
1071 \RequirePackage{zref-thepage}[2011/03/18]
1072 \RequirePackage{zref-lastpage}[2011/03/18]
1073 \RequirePackage{uniquecounter}[2009/12/18]

1074 \UniqueCounterNew{znextpage}
1075
1076 \newcommand*{\znextpagesetup}{%
1077   \afterassignment\ZREF@np@setup@i
1078   \def\ZREF@np@call@unknown##1%
1079 }
1080 \def\ZREF@np@setup@i{%
1081   \afterassignment\ZREF@np@setup@ii
1082   \def\ZREF@np@call@nonext##1%
1083 }
1084 \def\ZREF@np@setup@ii{%
1085   \def\ZREF@np@call@next##1%
1086 }
1087 \def\ZREF@np@call@unknown#1{#1}
1088 \def\ZREF@np@call@nonext#1{#1}
1089 \def\ZREF@np@call@next#1{#1}
1090 \ZREF@ifDefinable\znextpage\def{%
1091   {\UniqueCounterCall{znextpage}{\ZREF@nextpage}}%
1092 }%
1093 \newcommand*{\znonextpagename}{}
1094 \newcommand*{\zunknownnextpagename}{\Z@D@page}
1095 \def\ZREF@nextpage#1{%
1096   \begingroup
1097     \def\ZREF@refname@this{zref@np#1}%
1098     \zref@labelbyprops\ZREF@refname@this{abspage}%
1099     \chardef\ZREF@call=0 % unknown
1100     \ZREF@ifrefundefined\ZREF@refname@this{%
1101       }{%
1102         \edef\ZREF@pagenum@this{%
1103           \zref@extractdefault\ZREF@refname@this{abspage}{0}%
1104         }%
1105         \edef\ZREF@refname@next{%
1106           \zref@thepage@name{%
1107             \the\numexpr\ZREF@pagenum@this+1%
1108           }%
1109         }%
1110         \ifnum\ZREF@pagenum@this>0 %
```

```

1111     \ZREF@ifrefundefined{LastPage}{%
1112     \zref@ifrefundefined\ZREF@refname@next{%
1113     }{%
1114     \chardef\ZREF@call=2 % next page
1115     }%
1116   }{%
1117     \edef\ZREF@pagenum@last{%
1118     \zref@extractdefault{LastPage}{abspage}{0}%
1119     }%
1120     \ifnum\ZREF@pagenum@this<\ZREF@pagenum@last\ltx@space
1121     \ZREF@ifrefundefined\ZREF@refname@next{%
1122     }{%
1123     \chardef\ZREF@call=2 % next page
1124     }%
1125     \else
1126     \ifnum\ZREF@pagenum@this=\ZREF@pagenum@this\ltx@space
1127     \chardef\ZREF@call=1 % no next page
1128     \fi
1129     \fi
1130   }%
1131 \fi
1132 }%
1133 \edef\x{%
1134 \endgroup
1135 \ifcase\ZREF@call
1136 \noexpand\ZREF@np@call@unknown{%
1137 \noexpand\zunknownnextpagename
1138 }%
1139 \or
1140 \noexpand\ZREF@np@call@nonext{%
1141 \noexpand\znonextpagename
1142 }%
1143 \else
1144 \noexpand\ZREF@np@call@next{%
1145 \noexpand\zref@extract{\ZREF@refname@next}{page}%
1146 }%
1147 \fi
1148 }%
1149 \x
1150 }
1151 </nextpage>

```

## 6.9 Module `totpages`

```

1152 <*totpages>
1153 \NeedsTeXFormat{LaTeX2e}
1154 \ProvidesPackage{zref-totpages}%
1155 [2011/03/18 v2.21 Module totpages for zref (HO)]%
1156 \RequirePackage{zref-base}[2011/03/18]
1157 \ifx\ZREF@base@ok Y%
1158 \else
1159 \expandafter\endinput
1160 \fi

```

The absolute page number of the last page is the total page number.

```

1161 \RequirePackage{zref-abspage}[2011/03/18]
1162 \RequirePackage{zref-lastpage}[2011/03/18]

```

`\ztotpages` Macro `\ztotpages` contains the number of pages. It can be used inside expandable calculations. It expands to zero if the reference is not yet available.

```

1163 \newcommand*{\ztotpages}{%
1164 \zref@extractdefault{LastPage}{abspage}{0}%
1165 }

```

Also we mark the reference LastPage as used:

```
1166 \AtBeginDocument{%
1167   \zref@refused{LastPage}%
1168 }
1169 </totpages>
```

## 6.10 Module pagelayout

```
1170 <*pagelayout>
1171 \NeedsTeXFormat{LaTeX2e}
1172 \ProvidesPackage{zref-pagelayout}%
1173 [2011/03/18 v2.21 Module pagelayout for zref (HO)]%
1174 \RequirePackage{zref-base}[2011/03/18]
1175 \ifx\ZREF@base@ok Y%
1176 \else
1177   \expandafter\endinput
1178 \fi

1179 \RequirePackage{zref-thepage}[2011/03/18]
1180 \RequirePackage{ifluatex}[2010/03/01]
1181 \RequirePackage{atveryend}[2010/03/24]

1182 \ifluatex
1183   \ifnum\luatexversion<39 %
1184   \else
1185     \begingroup
1186       \escapechar=-1 %
1187       \def\ZREF@temp#1{%
1188         \ltx@ifundefined{\string#1}{%
1189           \let#1\ltx@undefined
1190           \directlua{%
1191             if tex.enableprimitives then %
1192               tex.enableprimitives('', {\string#1})%
1193             end%
1194           }%
1195           \ltx@ifundefined{\string#1}{%
1196             }{%
1197               \global#1=#1%
1198               \@PackageInfoNoLine{zref-pagelayout}{%
1199                 \string#1 enabled%
1200               }%
1201             }%
1202           }{%
1203             }%
1204             \ZREF@temp\pdfpagewidth
1205             \ZREF@temp\pdfpageheight
1206             \ZREF@temp\pdfhorigin
1207             \ZREF@temp\pdfvorigin
1208           \endgroup
1209         \fi
1210       \fi

1211       \def\ZREF@temp#1{%
1212         \begingroup
1213           \escapechar=-1 %
1214           \ltx@ifundefined{\string#1}{\endgroup}{%
1215             \edef\x{%
1216               \endgroup
1217               \noexpand\zref@newprop*{\string#1}%
1218                 [\number\noexpand#1]% hash-ok
1219                 {\number\noexpand#1}%
1220               \noexpand\zref@addprop{thepage}{\string#1}%
1221             }%
1222             \x
```

```

1223 }%
1224 }
1225 \ZREF@temp\mag
1226 \ZREF@temp\paperwidth
1227 \ZREF@temp\paperheight
1228 \ZREF@temp\stockwidth
1229 \ZREF@temp\stockheight
1230 \ZREF@temp\pdfpagewidth
1231 \ZREF@temp\pdfpageheight
1232 \ZREF@temp\pdfhorigin
1233 \ZREF@temp\pdfvorigin
1234 \ZREF@temp\hoffset
1235 \ZREF@temp\voffset
1236 \ZREF@temp\topmargin
1237 \ZREF@temp\oddsidemargin
1238 \ZREF@temp\evensidemargin
1239 \ZREF@temp\textwidth
1240 \ZREF@temp\textheight
1241 \ZREF@temp\headheight
1242 \ZREF@temp\headsep
1243 \ZREF@temp\footskip
1244 \ZREF@temp\marginparwidth
1245 \ZREF@temp\marginparsep
1246 \ZREF@temp\columnwidth
1247 \ZREF@temp\columnsep

```

\ifZREF@pl@list

```
1248 \ltx@newif\ifZREF@pl@list
```

\zref@listpagelayout

```

1249 \ZREF@IfDefinable\zlistpagelayout\def{%
1250   {\global\ZREF@pl@listtrue}%
1251 }

```

\ZREF@pl@AfterLastShipout

```

1252 \def\ZREF@pl@AfterLastShipout{%
1253   \ifZREF@pl@list
1254     \edef\ZREF@page@max{\the\value{abspage}}%
1255     \ltx@ifundefined{ZREF@org@testdef}{%
1256       \let\ZREF@org@testdef\@testdef
1257       \def\@testdef##1##2##3{%
1258         \ZREF@org@testdef{##1}{##2}{##3}%
1259       \def\ZREF@temp{##1}%
1260       \ifx\ZREF@temp\ZREF@RefPrefix
1261         \expandafter\gdef\csname##1@##2\endcsname{##3}%
1262       \fi
1263     }%
1264   }{}%
1265   \AtVeryEndDocument{\ZREF@pl@AtVeryEnd}%
1266   \fi
1267 }

```

\ZREF@pl@AtVeryEnd

```

1268 \def\ZREF@pl@AtVeryEnd{%
1269   \begingroup
1270   \toks@{Page layout parameters:\MessageBreak}%
1271   \count@=1 %
1272   \ZREF@pl@ListPage
1273   \edef\x{\endgroup
1274     \noexpand\@PackageInfoNoLine{zref-pagelayout}{\the\toks@}%
1275   }%
1276   \x
1277 }

```

\ZREF@pl@ListPage

```
1278 \def\ZREF@pl@ListPage{%
1279   \edef\x{%
1280     \toks@={%
1281       \the\toks@
1282       Page \the\count@:\noexpand\MessageBreak
1283       \zref@ifrefundefined{thepage\the\count@}{-}{%
1284         \ltx@space\ltx@space mag = %
1285         \zref@extract{thepage\the\count@}{mag}%
1286         \noexpand\MessageBreak
1287         \ZREF@pl@ListEntry{paperwidth}%
1288         \ZREF@pl@ListEntry{paperheight}%
1289         \ZREF@pl@ListEntry{stockwidth}%
1290         \ZREF@pl@ListEntry{stockheight}%
1291         \ZREF@pl@ListEntry{pdfpagewidth}%
1292         \ZREF@pl@ListEntry{pdfpageheight}%
1293         \ZREF@pl@ListEntry{pdfhorigin}%
1294         \ZREF@pl@ListEntry{pdfvorigin}%
1295         \ZREF@pl@ListEntry{hoffset}%
1296         \ZREF@pl@ListEntry{voffset}%
1297         \ZREF@pl@ListEntry{topmargin}%
1298         \ZREF@pl@ListEntry{oddsidemargin}%
1299         \ZREF@pl@ListEntry{evensidemargin}%
1300         \ZREF@pl@ListEntry{textwidth}%
1301         \ZREF@pl@ListEntry{textheight}%
1302         \ZREF@pl@ListEntry{headheight}%
1303         \ZREF@pl@ListEntry{headsep}%
1304         \ZREF@pl@ListEntry{footskip}%
1305         \ZREF@pl@ListEntry{marginparwidth}%
1306         \ZREF@pl@ListEntry{marginparsep}%
1307         \ZREF@pl@ListEntry{columnwidth}%
1308         \ZREF@pl@ListEntry{columnsep}%
1309       }%
1310     }%
1311   }\x
1312   \ifnum\ZREF@page@max>\count@
1313     \advance\count@ by\ltx@one
1314   \else
1315     \expandafter\ltx@gobble
1316   \fi
1317   \ZREF@pl@ListPage
1318 }
```

\ZREF@pl@ListEntry

```
1319 \def\ZREF@pl@ListEntry#1{%
1320   \zref@ifpropundefined{#1}{-%
1321   }{%
1322     \zref@ifrefcontainsprop{thepage\the\count@}{#1}{%
1323       \ltx@space\ltx@space#1 = %
1324       \zref@extract{thepage\the\count@}{#1}sp = %
1325       \the\dimexpr\zref@extract{thepage\the\count@}{#1}sp\relax
1326       \noexpand\MessageBreak
1327     }{}%
1328   }%
1329 }

1330 \AfterLastShipout{%
1331   \ZREF@pl@AfterLastShipout
1332 }

1333 </pagelayout>
```

## 6.11 Module pageattr

```
1334 <*pageattr>
1335 \NeedsTeXFormat{LaTeX2e}
1336 \ProvidesPackage{zref-pageattr}%
1337 [2011/03/18 v2.21 Module pageattr for zref (HO)]%
1338 \RequirePackage{zref-base}[2011/03/18]
1339 \ifx\ZREF@base@ok Y%
1340 \else
1341 \expandafter\endinput
1342 \fi

1343 \RequirePackage{ifluatex}[2010/03/01]

1344 \ifluatex
1345 \ifnum\luatexversion<39 %
1346 \else
1347 \begingroup
1348 \escapechar=-1 %
1349 \def\ZREF@temp#1{%
1350 \ltx@ifundefined{\string#1}{%
1351 \let#1\ltx@undefined
1352 \directlua{%
1353 if tex.enableprimitives then %
1354 tex.enableprimitives('', {\string#1})%
1355 end%
1356 }%
1357 \ltx@ifundefined{\string#1}{%
1358 }{%
1359 \global#1=#1%
1360 \@PackageInfoNoLine{zref-pageattr}{%
1361 \string#1 enabled%
1362 }%
1363 }%
1364 }{}%
1365 }%
1366 \ZREF@temp\pdfpageattr
1367 \ZREF@temp\pdfpagesattr
1368 \endgroup
1369 \fi
1370 \fi

1371 \let\ZREF@temp=N%
1372 \ltx@ifundefined{pdfpageattr}{%
1373 \@PackageInfoNoLine{zref-pageattr}{%
1374 \string\pdfpageattr\space is not available%
1375 }%
1376 \def\zref@pdfpageattr#1{}%
1377 \def\zref@pdfpageattr@used#1{}%
1378 }{}%
1379 \RequirePackage{zref-thepage}[2011/03/18]%
1380 \zref@newprop*{pdfpageattr}[]{\zref@hex{\the\pdfpageattr}}%
1381 \zref@addprop{thepage}{pdfpageattr}%
1382 \let\ZREF@temp=Y%
1383 }

1384 \ltx@ifundefined{pdfpagesattr}{%
1385 \@PackageInfoNoLine{zref-pageattr}{%
1386 \string\pdfpagesattr\space is not available%
1387 }%
1388 \def\zref@pdfpagesattr{}%
1389 \def\zref@pdfpagesattr@used{}%
1390 }{}%
1391 \RequirePackage{zref-lastpage}[2011/03/18]%
1392 \zref@newprop*{pdfpagesattr}[]{\zref@hex{\the\pdfpagesattr}}%
1393 \zref@addprop{LastPage}{pdfpagesattr}%
```

```

1394 \let\ZREF@temp=Y%
1395 }%

1396 \ifx\ZREF@temp N%
1397 \expandafter\endinput
1398 \fi

1399 \RequirePackage{zref-abspage}[2011/03/18]
1400 \RequirePackage{atveryend}[2010/03/24]
1401 \RequirePackage{pdftexcmds}[2010/04/01]
1402 \let\ZREF@temp=Y%
1403 \ltx@ifundefined{pdf@escapehex}{\let\ZREF@temp=N}{%}
1404 \ltx@ifundefined{pdf@unescapehex}{\let\ZREF@temp=N}{%}
1405 \ifx\ZREF@temp N%
1406 \let\zref@hex\ltx@firstofone
1407 \let\zref@unhex\ltx@firstofone
1408 \else
1409 \let\zref@hex\pdf@escapehex
1410 \let\zref@unhex\pdf@unescapehex
1411 \fi

```

\ifZREF@pa@list

```

1412 \ltx@newif\ifZREF@pa@list

```

\zref@listpageattr

```

1413 \ZREF@ifdefinable\zlistpageattr\def{%
1414   {\ZREF@pa@listtrue}%
1415 }

```

\ZREF@pa@AfterLastShipout

```

1416 \def\ZREF@pa@AfterLastShipout{%
1417   \ifZREF@pa@list
1418     \edef\ZREF@page@max{\the\value{abspage}}%
1419     \ltx@ifundefined{ZREF@org@testdef}{%
1420       \let\ZREF@org@testdef\@testdef
1421       \def\@testdef##1##2##3{%
1422         \ZREF@org@testdef{##1}{##2}{##3}%
1423         \def\ZREF@temp{##1}%
1424         \ifx\ZREF@temp\ZREF@RefPrefix
1425           \expandafter\xdef\csname##1@##2\endcsname{##3}%
1426         \fi
1427       }%
1428     }{%
1429     \AtVeryEndDocument{\ZREF@pa@AtVeryEnd}%
1430   \fi
1431 }

```

\ZREF@pa@AtVeryEnd

```

1432 \ltx@ifundefined{pdfpageattr}{%
1433   \def\ZREF@pa@AtVeryEnd{%
1434   }{%
1435   \def\ZREF@pa@AtVeryEnd{%
1436     \begingroup
1437       \toks@{List of \ltx@backslashchar pdfpageattr:\MessageBreak}%
1438       \count@=1 %
1439       \ZREF@pa@ListPage
1440       \edef\x{\endgroup
1441         \noexpand\@PackageInfoNoLine{zref-pageattr}{%
1442           \the\toks@
1443         }%
1444       }%
1445     \x
1446   }%

```



\zref@pageattr

```
1447 \def\zref@pageattr#1{%
1448   \zref@unhex{%
1449     \zref@extract{thepage\ZREF@number#1}{pdfpageattr}%
1450   }%
1451 }
```

\zref@pageattr@used

```
1452 \ZREF@Robust\def\zref@pageattr@used#1{%
1453   \zref@refused{thepage\ZREF@number#1}%
1454 }
```

\ZREF@pa@ListPage

```
1455 \def\ZREF@pa@ListPage{%
1456   \edef\x{%
1457     \toks@={%
1458       \the\toks@
1459       Page \the\count@:%
1460       \noexpand\MessageBreak
1461       \zref@ifrefundefined{thepage\the\count@}{}{%
1462         <<\zref@pdfpageattr\count@>>%
1463       \noexpand\MessageBreak
1464     }%
1465   }%
1466   }\x
1467   \ifnum\ZREF@page@max>\count@
1468     \advance\count@ by\ltx@one
1469   \else
1470     \expandafter\ltx@gobble
1471   \fi
1472   \ZREF@pa@ListPage
1473 }%
1474 }

1475 \ltx@ifundefined{pdfpagesattr}{%
1476 }{%
```

\zref@pdfpagesattr

```
1477 \def\zref@pdfpagesattr{%
1478   \zref@unhex{%
1479     \zref@extract{LastPage}{pdfpagesattr}%
1480   }%
1481 }
```

\zref@pdfpagesattr@used

```
1482 \ZREF@Robust\def\zref@pdfpagesattr@used{%
1483   \zref@refused{LastPage}%
1484 }%

1485 \ltx@LocalAppendToMacro\ZREF@pa@AtVeryEnd{%
1486   \@PackageInfoNoLine{zref-pageattr}{%
1487     \ltx@backslashchar pdfpagesattr:\MessageBreak
1488     <<\zref@pdfpagesattr>>%
1489     \MessageBreak
1490   }%
1491 }%
1492 }

1493 \AfterLastShipout{%
1494   \ZREF@pa@AfterLastShipout
1495 }

1496 </pageattr>
```

## 6.12 Module marks

```
1497 <*marks>
1498 \NeedsTeXFormat{LaTeX2e}
1499 \ProvidesPackage{zref-marks}%
1500 [2011/03/18 v2.21 Module marks for zref (HO)]%
1501 \RequirePackage{zref-base}[2011/03/18]
1502 \ifx\ZREF@base@ok Y%
1503 \else
1504 \expandafter\endinput
1505 \fi

1506 \newcommand*{\zref@marks@register}[3][ ]{%
1507 \edef\ZREF@TempName{#1}%
1508 \edef\ZREF@TempNum{\ZREF@number#2}%
1509 \ifnum\ZREF@TempNum<\ltx@zero %
1510 \PackageError\ZREF@name{%
1511 \string\zref@marks@register\ltx@space is called with invalid%
1512 \MessageBreak
1513 marks register number (\ZREF@TempNum)%
1514 }{%
1515 Use '0' or the command, defined by \string\newmarks.\MessageBreak
1516 \@ehc
1517 }%
1518 \else
1519 \ifx\ZREF@TempName\ltx@empty
1520 \edef\ZREF@TempName{mark\romannumeral\ZREF@TempNum}%
1521 \else
1522 \edef\ZREF@TempName{marks\ZREF@TempName}%
1523 \fi
1524 \ZREF@MARKS@DefineProp{top}%
1525 \ZREF@MARKS@DefineProp{first}%
1526 \ZREF@MARKS@DefineProp{bot}%
1527 \kv@parse{#3}{%
1528 \ifx\kv@value\relax
1529 \def\kv@value{top,first,bot}%
1530 \fi
1531 \edef\ZREF@temp{\expandafter\ltx@car\kv@key X\@nil}%
1532 \ifx\ZREF@temp\ZREF@STAR
1533 \edef\kv@key{\expandafter\ltx@cdr\kv@key\@nil}%
1534 \zref@newlist\kv@key
1535 \fi
1536 \expandafter\comma@parse\expandafter{\kv@value}{%
1537 \ifcase0\ifx\comma@entry\ZREF@NAME@top 1\else
1538 \ifx\comma@entry\ZREF@NAME@first 1\else
1539 \ifx\comma@entry\ZREF@NAME@bot 1\fi\fi\fi\ltx@space
1540 \PackageWarning{zref-marks}{%
1541 Use 'top', 'first' or 'bot' for the list values%
1542 \MessageBreak
1543 in the third argument of \string\zref@marks@register.%
1544 \MessageBreak
1545 Ignoring unkown value '\comma@entry'%
1546 }%
1547 \else
1548 \zref@addprop{\kv@key}{\comma@entry\ZREF@TempName}%
1549 \fi
1550 \ltx@gobble
1551 }%
1552 \ltx@gobbletwo
1553 }%
1554 \fi
1555 }
1556 \def\ZREF@STAR{*}
1557 \def\ZREF@NAME@top{top}
```

```

1558 \def\ZREF@NAME@first{first}
1559 \def\ZREF@NAME@bot{bot}
1560 \def\ZREF@MARKS@DefineProp#1{%
1561   \zref@ifpropundefined{#1\ZREF@TempName}{%
1562     \ifnum\ZREF@TempNum=\ltx@zero
1563       \begingroup
1564         \edef\x{\endgroup
1565           \noexpand\zref@newprop*{#1\ZREF@TempName}[]{%
1566             \expandafter\noexpand\csname#1mark\endcsname
1567           }%
1568         }%
1569       \x
1570     \else
1571       \begingroup
1572         \edef\x{\endgroup
1573           \noexpand\zref@newprop*{#1\ZREF@TempName}[]{%
1574             \expandafter\noexpand\csname#1marks\endcsname
1575             \ZREF@TempNum
1576           }%
1577         }%
1578       \x
1579     \fi
1580 }{%
1581   \PackageWarning{zref-marks}{%
1582     \string\zref@marks@register\ltx@space does not generate the%
1583     \MessageBreak
1584     new property ‘#1\ZREF@TempName’, because\MessageBreak
1585     it is already defined%
1586   }%
1587 }%
1588 }
1589 </marks>

```

### 6.13 Module runs

This module does not use the label-reference-system. The reference changes with each L<sup>A</sup>T<sub>E</sub>X run and would force a rerun warning always.

```

1590 <*runs>
1591 \NeedsTeXFormat{LaTeX2e}
1592 \ProvidesPackage{zref-runs}%
1593 [2011/03/18 v2.21 Module runs for zref (HO)]%

```

\zruns

```

1594 \providecommand*\zruns{0}%
1595 \AtBeginDocument{%
1596   \edef\zruns{\number\numexpr\zruns+1}%
1597   \begingroup
1598     \def\on@line{}%
1599     \PackageInfo{zref-runs}{LaTeX runs: \zruns}%
1600     \if@filesw
1601       \immediate\write\@mainaux{%
1602         \string\gdef\string\zruns{\zruns}%
1603       }%
1604     \fi
1605   \endgroup
1606 }
1607 </runs>

```

### 6.14 Module perpage

```

1608 <*perpage>

```

```

1609 \NeedsTeXFormat{LaTeX2e}
1610 \ProvidesPackage{zref-perpage}%
1611 [2011/03/18 v2.21 Module perpage for zref (HO)]%
1612 \RequirePackage{zref-base}[2011/03/18]
1613 \ifx\ZREF@base@ok Y%
1614 \else
1615 \expandafter\endinput
1616 \fi

```

This module resets a counter at page boundaries. Because of the asynchronous output routine page counter properties cannot be asked directly, references are necessary.

For detecting changed pages module `abspage` is loaded.

```

1617 \RequirePackage{zref-abspage}[2011/03/18]

```

We group the properties for the needed references in the property list `perpage`. The property `pagevalue` records the correct value of the page counter.

```

1618 \zref@newprop*{pagevalue}[0]{\number\c@page}
1619 \zref@newlist{perpage}
1620 \zref@addprops{perpage}{abspage,page,pagevalue}

```

The page value, known by the reference mechanism, will be stored in counter `zpage`.

```

1621 \newcounter{zpage}

```

Counter `zref@unique` helps in generating unique reference names.

```

1622 \zref@require@unique

```

In order to be able to reset the counter, we hook here into `\stepcounter`. In fact two nested hooks are used to allow other packages to use the first hook at the beginning of `\stepcounter`.

```

1623 \let\ZREF@org@stepcounter\stepcounter
1624 \def\stepcounter#1{%
1625   \ifcsname @stepcounterhook@#1\endcsname
1626     \csname @stepcounterhook@#1\endcsname
1627   \fi
1628   \ZREF@org@stepcounter{#1}%
1629 }

```

`\zmakeperpage` Makro `\zmakeperpage` resets a counter at each page break. It uses the same syntax and semantics as `\MakePerPage` from package `perpage` [5]. The initial start value can be given by the optional argument. Default is one that means after the first `\stepcounter` on a new page the counter starts with one.

```

1630 \ZREF@IfDefinable\zmakeperpage\def{%
1631   {%
1632     \ifnextchar[\ZREF@makeperpage@opt{\ZREF@@makeperpage[\ltx@zero]}%
1633   }%
1634 }

```

We hook before the counter is incremented in `\stepcounter`, package `perpage` afterwards. Thus a little calculation is necessary.

```

1635 \def\ZREF@makeperpage@opt[#1]{%
1636   \begingroup
1637   \edef\x{\endgroup
1638     \noexpand\ZREF@@makeperpage[\number\numexpr#1-1\relax]}%
1639   }%
1640   \x
1641 }

1642 \def\ZREF@@makeperpage[#1]#2{%
1643   \ifundefined{@stepcounterhook@#2}{%
1644     \expandafter\gdef\csname @stepcounterhook@#2\endcsname{}%
1645   }{%
1646     \expandafter\gdef\csname ZREF@perpage@#2\endcsname{%
1647       \ZREF@@perpage@step{#2}{#1}%
1648     }%

```

```

1649 \expandafter\g@addto@macro\csname @stepcounterhook@#2\endcsname{%
1650   \ifcsname ZREF@perpage@#2\endcsname
1651     \csname ZREF@perpage@#2\endcsname
1652   \fi
1653 }%
1654 }

```

`\ZREF@@perpage@step` The heart of this module follows.

```

1655 \def\ZREF@@perpage@step#1#2{%

```

First the reference is generated.

```

1656 \global\advance\c@zref@unique\ltx@one
1657 \begingroup
1658 \expandafter
1659 \zref@labelbylist\expandafter{\thezref@unique}{perpage}%

```

The `\expandafter` commands are necessary, because `\ZREF@temp` is also used inside of `\zref@labelbylist`.

The evaluation of the reference follows. If the reference is not yet known, we use the page counter as approximation.

```

1660 \zref@ifrefundefined\thezref@unique{%
1661   \global\c@zpage=\c@page
1662   \global\let\thezpage\thepage
1663   \expandafter\xdef\csname ZREF@abspage@#1\endcsname{%
1664     \number\c@abspage
1665   }%
1666 }{%

```

The reference is used to set `\thezpage` and counter `zpage`.

```

1667   \global\c@zpage=\zref@extract\thezref@unique{pagevalue}\relax
1668   \xdef\thezpage{\noexpand\zref@extract{\thezref@unique}{page}}%
1669   \expandafter\xdef\csname ZREF@abspage@#1\endcsname{%
1670     \zref@extractdefault\thezref@unique
1671     {abspage}{\number\c@abspage}%
1672   }%
1673 }%

```

Page changes are detected by a changed absolute page number.

```

1674 \expandafter\ifx\csname ZREF@abspage@#1\endcsname
1675   \csname ZREF@currentabspage@#1\endcsname
1676 \else
1677   \global\csname c@#1\endcsname=#2\relax
1678   \global\expandafter\let
1679     \csname ZREF@currentabspage@#1\endcsname
1680     \csname ZREF@abspage@#1\endcsname
1681 \fi
1682 \endgroup
1683 }

```

`\zunmakeperpage` Macro `\zunmakeperpage` cancels the effect of `\zmakeperpage`.

```

1684 \ZREF@ifDefinable\zunmakeperpage\def{%
1685   #1{%
1686     \global\expandafter
1687     \let\csname ZREF@perpage@#1\endcsname\@undefined
1688   }%
1689 }

```

```

1690 </perpage>

```

## 6.15 Module `titleref`

```

1691 <*titleref>
1692 \NeedsTeXFormat{LaTeX2e}

```

```

1693 \ProvidesPackage{zref-titleref}%
1694 [2011/03/18 v2.21 Module titleref for zref (HO)]%
1695 \RequirePackage{zref-base}[2011/03/18]
1696 \ifx\ZREF@base@ok Y%
1697 \else
1698 \expandafter\endinput
1699 \fi
1700 \RequirePackage{getttitlestring}[2009/12/08]

```

### 6.15.1 Implementation

```
1701 \RequirePackage{keyval}
```

This module makes section and caption titles available for the reference system. It uses some of the ideas of package `nameref` and `titleref`.

Now we can add the property `title` is added to the main property list.

```

1702 \ZREF@NewPropTitle
1703 \zref@addprop\ZREF@mainlist{title}%

```

The title strings go into the `.aux` file, thus they need some kind of protection. Package `titleref` uses a protected expansion method. The advantage is that this can be used to cleanup the string and to remove `\label`, `\index` and other macros unwanted for referencing. But there is the risk that fragile stuff can break.

Therefore package `nameref` does not expand the string. Thus the entries can safely be written to the `.aux` file. But potentially dangerous macros such as `\label` remain in the string and can cause problems when using the string in references.

`\ifzref@titleref@expand` The switch `\ifzref@titleref@expand` distinguishes between the both methods. Package `nameref`'s behaviour is achieved by setting the switch to false, otherwise `titleref`'s expansion is used. Default is false.

```
1704 \newif\ifzref@titleref@expand
```

`\ZREF@titleref@hook` The hook `\ZREF@titleref@hook` allows to extend the cleanup for the expansion method. Thus unnecessary macros can be removed or dangerous commands removed. The hook is executed before the expansion of `\zref@titleref@current`.

```
1705 \let\ZREF@titleref@hook\ltx@empty
```

`\zref@titleref@cleanup` The hook should not be used directly, instead we provide the macro `\zref@titleref@cleanup` to add stuff to the hook and prevents that a previous non-empty content is not discarded accidentally.

```

1706 \ZREF@Robust\def\zref@titleref@cleanup#1{%
1707   \begingroup
1708   \toks@\expandafter{%
1709     \ZREF@titleref@hook
1710     #1%
1711   }%
1712   \expandafter\endgroup
1713   \expandafter\def\expandafter\ZREF@titleref@hook\expandafter{%
1714     \the\toks@
1715   }%
1716 }%

```

`\ifzref@titleref@stripperiod` Sometimes a title contains a period at the end. Package `nameref` removes this. This behaviour is controlled by the switch `\ifzref@titleref@stripperiod` and works regardless of the setting of option `expand`. Period stripping is the default.

```

1717 \newif\ifzref@titleref@stripperiod
1718 \zref@titleref@stripperiodtrue

```

`\zref@titleref@setcurrent` Macro `\zref@titleref@setcurrent` sets a new current title stored in `\zref@titleref@current`. Some cleanup and expansion is performed that can be controlled by the previous switches.

```

1719 \ZREF@Robust\def\zref@titleref@setcurrent#1{%
1720   \ifzref@titleref@expand

```

```

1721   \GetTitleStringExpand{#1}%
1722 \else
1723   \GetTitleStringNonExpand{#1}%
1724 \fi
1725 \edef\zref@titleref@current{%
1726   \detokenize\expandafter{\GetTitleStringResult}%
1727 }%
1728 \ifzref@titleref@stripperperiod
1729   \edef\zref@titleref@current{%
1730     \expandafter\ZREF@stripperperiod\zref@titleref@current
1731     \ltx@empty.\ltx@empty\@nil
1732   }%
1733 \fi
1734 }%
1735 \GetTitleStringDisableCommands{%
1736   \ZREF@titleref@hook
1737 }

```

`\ZREF@stripperperiod` If `\ZREF@stripperperiod` is called, the argument consists of space tokens and tokens with catcode 12 (other), because of  $\varepsilon$ -TeX's `\detokenize`.

```
1738 \def\ZREF@stripperperiod#1.\ltx@empty#2\@nil{#1}%
```

### 6.15.2 User interface

`\ztitlerefsetup` The behaviour of module `titleref` is controlled by switches and a hook. They can be set by `\ztitlerefsetup` with a key value interface, provided by package `keyval`. Also the current title can be given explicitly by the key `title`.

```

1739 \define@key{ZREF@TR}{expand}[true]{%
1740   \csname zref@titleref@expand#1\endcsname
1741 }%
1742 \define@key{ZREF@TR}{stripperperiod}[true]{%
1743   \csname zref@titleref@stripperperiod#1\endcsname
1744 }%
1745 \define@key{ZREF@TR}{cleanup}{%
1746   \zref@titleref@cleanup{#1}%
1747 }%
1748 \define@key{ZREF@TR}{title}{%
1749   \def\zref@titleref@current{#1}%
1750 }%
1751 \ZREF@IfDefinable\ztitlerefsetup\def{%
1752   {\kvsetkeys{ZREF@TR}}%
1753 }%

```

`\ztitleref` The user command `\ztitleref` references the title. For safety `\label` is disabled to prevent multiply defined references.

```

1754 \ZREF@IfDefinable\ztitleref\def{%
1755   {\zref@wrapper@babel\ZREF@titleref}%
1756 }%
1757 \def\ZREF@titleref#1{%
1758   \begingroup
1759     \zref@refused{#1}%
1760     \let\label\ltx@gobble
1761     \zref@extract{#1}{title}%
1762   \endgroup
1763 }%

```

### 6.15.3 Patches for section and caption commands

The section and caption macros are patched to extract the title data.

Captions of figures and tables.

```
1764 \AtBeginDocument{%
```

```

1765 \ZREF@patch{@caption}{%
1766   \long\def\@caption#1[#2]{%
1767     \zref@titleref@setcurrent{#2}%
1768     \ZREF@org@@caption{#1}[#2]}%
1769   }%
1770 }%

```

Section commands without star. The title version for the table of contents is used because it is usually shorter and more robust.

```

1771 \ZREF@patch{@part}{%
1772   \def\@part[#1]{%
1773     \zref@titleref@setcurrent{#1}%
1774     \ZREF@org@@part[#{#1}]%
1775   }%
1776 }%
1777 \ZREF@patch{@chapter}{%
1778   \def\@chapter[#1]{%
1779     \zref@titleref@setcurrent{#1}%
1780     \ZREF@org@@chapter[#{#1}]%
1781   }%
1782 }%
1783 \ZREF@patch{@sect}{%
1784   \def\@sect#1#2#3#4#5#6[#7]{%
1785     \zref@titleref@setcurrent{#7}%
1786     \ZREF@org@@sect{#1}{#2}{#3}{#4}{#5}{#6}[#{#7}]%
1787   }%
1788 }%

```

The star versions of the section commands.

```

1789 \ZREF@patch{@spart}{%
1790   \def\@spart#1{%
1791     \zref@titleref@setcurrent{#1}%
1792     \ZREF@org@@spart{#1}%
1793   }%
1794 }%
1795 \ZREF@patch{@schapter}{%
1796   \def\@schapter#1{%
1797     \zref@titleref@setcurrent{#1}%
1798     \ZREF@org@@schapter{#1}%
1799   }%
1800 }%
1801 \ZREF@patch{@ssect}{%
1802   \def\@ssect#1#2#3#4#5{%
1803     \zref@titleref@setcurrent{#5}%
1804     \ZREF@org@@ssect{#1}{#2}{#3}{#4}{#5}%
1805   }%
1806 }%

```

#### 6.15.4 Environment description

```

1807 \ZREF@patch{descriptionlabel}{%
1808   \def\descriptionlabel#1{%
1809     \zref@titleref@setcurrent{#1}%
1810     \ZREF@org@descriptionlabel{#1}%
1811   }%
1812 }%

```

#### 6.15.5 Class memoir

```

1813 \@ifclassloaded{memoir}{%
1814   \ltx@ifundefined{ifheadnameref}{-}{%
1815     \def\@chapter[#1]#2{%
1816       \ltx@ifundefined{ch@pt@c}{-}{%
1817         \zref@titleref@setcurrent{#1}%

```



```

1818 }{%
1819   \ifx\ch@pt@c\ltx@empty
1820     \zref@titleref@setcurrent{#2}%
1821   \else
1822     \def\NR@temp{#1}%
1823     \ifx\NR@temp\ltx@empty
1824       \expandafter\zref@titleref@setcurrent
1825       \expandafter{\ch@pt@c}%
1826     \else
1827       \ifheadnameref
1828         \zref@titleref@setcurrent{#1}%
1829       \else
1830         \expandafter\zref@titleref@setcurrent
1831         \expandafter{\ch@pt@c}%
1832       \fi
1833     \fi
1834   \fi
1835 }%
1836 \ZREF@org@@chapter[{#1}]{#2}%
1837 }%
1838 \ZREF@patch{M@sect}{%
1839   \def\M@sect#1#2#3#4#5#6[#7][#8]{%
1840     \ifheadnameref
1841       \zref@titleref@setcurrent{#8}%
1842     \else
1843       \zref@titleref@setcurrent{#7}%
1844     \fi
1845     \ZREF@org@M@sect{#1}{#2}{#3}{#4}{#5}{#6}[{#7}][{#8}]%
1846   }%
1847 }%
1848 }%
1849 }-{}%

```

### 6.15.6 Class beamer

```

1850 \@ifclassloaded{beamer}{%
1851   \ZREF@patch{beamer@section}{%
1852     \long\def\beamer@section[#1]{%
1853       \zref@titleref@setcurrent{#1}%
1854       \ZREF@org@beamer@section[{#1}]%
1855     }%
1856   }%
1857   \ZREF@patch{beamer@subsection}{%
1858     \long\def\beamer@subsection[#1]{%
1859       \zref@titleref@setcurrent{#1}%
1860       \ZREF@org@beamer@subsection[{#1}]%
1861     }%
1862   }%
1863   \ZREF@patch{beamer@subsubsection}{%
1864     \long\def\beamer@subsubsection[#1]{%
1865       \zref@titleref@setcurrent{#1}%
1866       \ZREF@org@beamer@subsubsection[{#1}]%
1867     }%
1868   }%
1869 }-{}%

```

### 6.15.7 Package titlesec

```

1870 \@ifpackageloaded{titlesec}{%
1871   \ZREF@patch{ttl@sect@i}{%
1872     \def\ttl@sect@i#1#2[#3]#4{%
1873       \zref@titleref@setcurrent{#4}%
1874       \ZREF@org@ttl@sect@i{#1}{#2}[{#3}]{#4}%
1875     }%
1876   }%

```

```
1877 }{}%
```

### 6.15.8 Package longtable

Package longtable: some support for its \caption. However \label inside the caption is not supported.

```
1878 \@ifpackageloaded{longtable}{%
1879   \ZREF@patch{LT@c@ption}{%
1880     \def\LT@c@ption#1[#2]#3{%
1881       \ZREF@org@LT@c@ption{#1}[{#2}]{#3}%
1882       \zref@titleref@setcurrent{#2}%
1883     }%
1884   }%
1885 }{}%
```

### 6.15.9 Package listings

Package listings: support for its caption.

```
1886 \@ifpackageloaded{listings}{%
1887   \ZREF@patch{lst@MakeCaption}{%
1888     \def\lst@MakeCaption{%
1889       \ifx\lst@label\ltx@empty
1890       \else
1891         \expandafter\zref@titleref@setcurrent\expandafter{%
1892           \lst@caption
1893         }%
1894       \fi
1895     \ZREF@org@lst@MakeCaption
1896   }%
1897 }%
1898 }{}%
```

### 6.15.10 Theorems

```
1899 \ZREF@patch{@opargbegintheorem}{%
1900   \def\@opargbegintheorem#1#2#3{%
1901     \zref@titleref@setcurrent{#3}%
1902     \ZREF@org@@opargbegintheorem{#1}{#2}{#3}%
1903   }%
1904 }%
1905 \@ifpackageloaded{amsthm}{%
1906   \begingroup
1907     \edef\x{macro:\string#1\string#2[\string#3]}%
1908     \@onelevel@sanitize\x
1909     \def\y#1->#2\@nil{#1}%
1910     \edef\z{\expandafter\y\meaning\@begintheorem->\@nil}%
1911     \@onelevel@sanitize\z
1912     \expandafter\endgroup
1913     \ifx\x\z
1914       \ZREF@patch{@begintheorem}{%
1915         \def\@begintheorem#1#2[#3]{%
1916           \zref@titleref@setcurrent{#3}%
1917           \ZREF@org@@begintheorem{#1}{#2}[{#3}]%
1918         }%
1919       }%
1920     \fi
1921   }{}%
1922 }
1923 </titleref>
```

## 6.16 Module xr

```
1924 <*xr>
```

```

1925 \NeedsTeXFormat{LaTeX2e}
1926 \ProvidesPackage{zref-xr}%
1927 [2011/03/18 v2.21 Module xr for zref (HO)]%
1928 \RequirePackage{zref-base}[2011/03/18]
1929 \ifx\ZREF@base@ok Y%
1930 \else
1931 \expandafter\endinput
1932 \fi

1933 \RequirePackage{keyval}
1934 \RequirePackage{kvoptions}[2010/02/22]

```

We declare property `url`, because this is added, if a reference is imported and has not already set this field. Or if `hyperref` is used, then this property can be asked.

```

1935 \zref@newprop{url}{}%
1936 \zref@newprop{urluse}{}%
1937 \zref@newprop{externaldocument}{}%

```

Most code, especially the handling of the `.aux` files are taken from David Carlisle's `xr` package. Therefore I drop the documentation for these macros here.

`\zref@xr@ext` If the URL is not specied, then assume processed file with a guessed extension. Use the setting of `hyperref` if available.

```

1938 \providecommand*{\zref@xr@ext}{%
1939 \ltx@ifundefined{XR@ext}{pdf}{\XR@ext}%
1940 }%

```

`\ifZREF@xr@zreflabel` The use of the star form of `\zexternaldocument` is remembered in the switch `\ifZREF@xr@zreflabel`.

```

1941 \newif\ifZREF@xr@zreflabel

1942 \SetupKeyvalOptions{%
1943 family=ZREF@XR,%
1944 prefix=ZREF@xr%
1945 }
1946 \DeclareBoolOption[true]{tozreflabel}
1947 \DeclareBoolOption[false]{toltxlabel}
1948 \DeclareBoolOption[verbose]
1949 \define@key{ZREF@XR}{ext}{%
1950 \def\zref@xr@{#1}%
1951 }
1952 \DeclareBoolOption[false]{urluse}

```

`\zxrsetup`

```

1953 \newcommand*{\zxrsetup}{%
1954 \kvsetkeys{ZREF@XR}%
1955 }%

```

`\ZREF@xr@URL`

```

1956 \newcount\ZREF@xr@URL
1957 \ZREF@xr@URL=\ltx@zero

```

`\ZREF@xr@AddURL`

```

1958 \def\ZREF@xr@AddURL#1{%
1959 \begingroup
1960 \def\ZREF@temp{#1}%
1961 \count@=\ltx@one
1962 \ZREF@xr@@AddUrl
1963 \endgroup
1964 }

```

`\ZREF@xr@@AddUrl`

```
1965 \def\ZREF@xr@@AddUrl{%
1966   \ifnum\count@>\ZREF@xr@URL
1967     \global\advance\ZREF@xr@URL by\ltx@one
1968     \xdef\ZREF@xr@theURL{\romannumeral\ZREF@xr@URL}%
1969     \global\expandafter\let
1970     \csname Z@U@\ZREF@xr@theURL\endcsname\ZREF@temp
1971     \@PackageInfo{zref-xr}{%
1972       \ltx@backslashchar Z@U@\ZREF@xr@theURL:\MessageBreak
1973       \ZREF@temp\MessageBreak
1974     }%
1975   \else
1976     \expandafter
1977     \ifx\csname Z@U@\romannumeral\count@\endcsname\ZREF@temp
1978       \xdef\ZREF@xr@theURL{\romannumeral\count@}%
1979     \else
1980       \expandafter\expandafter\expandafter\ZREF@xr@@AddUrl
1981     \fi
1982   \fi
1983 }
```

`\zexternaldocument` In its star form it looks for `\newlabel`, otherwise for `\zref@newlabel`. Later we will read .aux files that expects @ to have catcode 11 (letter).

```
1984 \ZREF@IfDefinable\zexternaldocument\def{%
1985   {%
1986     \ZREF@NewPropAnchor
1987     \ZREF@NewPropTitle
1988     \begingroup
1989     \csname @safe@actives@true\endcsname
1990     \makeatletter
1991     \@ifstar{%
1992       \ZREF@xr@zreflabelfalse
1993       \@testopt\ZREF@xr@externaldocument{}%
1994     }{%
1995       \ZREF@xr@zreflabeltrue
1996       \@testopt\ZREF@xr@externaldocument{}%
1997     }%
1998   }%
1999 }
```

If the `\include` featur was used, there can be several .aux files. These files are read one after another, especially they are not recursively read in order to save read registers. Thus it can happen that the read order of the `newlabel` commands differs from L<sup>A</sup>T<sub>E</sub>X's order using `\input`.

`\ZREF@xr@externaldocument` It reads the remaining arguments. `\newcommand` comes in handy for the optional argument.

```
2000 \def\ZREF@xr@externaldocument[#1]#2{%
2001   \def\ZREF@xr@prefix{#1}%
2002   \let\ZREF@xr@filelist\ltx@empty
2003   \edef\ZREF@xr@externalfile{#2}%
2004   \edef\ZREF@xr@file{\ZREF@xr@externalfile.aux}%
2005   \filename@parse{#2}%
2006   \@testopt\ZREF@xr@graburl{#2.\zref@xr@ext}%
2007 }%
2008 \def\ZREF@xr@graburl[#1]{%
2009   \edef\ZREF@xr@url{#1}%
2010   \ifZREF@xr@urluse
2011     \expandafter\ZREF@xr@AddURL\expandafter{\ZREF@xr@url}%
2012     \expandafter\def\expandafter\ZREF@xr@url
2013     \expandafter{\csname Z@U@\ZREF@xr@theURL\endcsname}%
2014   \fi
```

```

2015 \ZREF@xr@checkfile
2016 \endgroup
2017 }%

```

\ZREF@xr@processfile We follow xr here, \IfFileExists offers a nicer test, but we have to open the file anyway.

```

2018 \def\ZREF@xr@checkfile{%
2019 \openin\@inputcheck\ZREF@xr@file\relax
2020 \ifeof\@inputcheck
2021 \PackageWarning{zref-xr}{%
2022 File '\ZREF@xr@file' not found or empty,\MessageBreak
2023 labels not imported%
2024 }%
2025 \else
2026 \PackageInfo{zref-xr}{%
2027 Label \ifZREF@xr@zreflabel (zref) \fi
2028 import from '\ZREF@xr@file'%
2029 }%
2030 \def\ZREF@xr@found{0}%
2031 \def\ZREF@xr@ignored@empty{0}%
2032 \def\ZREF@xr@ignored@zref{0}%
2033 \def\ZREF@xr@ignored@ltx{0}%
2034 \ZREF@xr@processfile
2035 \closein\@inputcheck
2036 \begingroup
2037 \let\on@line\ltx@empty
2038 \PackageInfo{zref-xr}{%
2039 Statistics for '\ZREF@xr@file':\MessageBreak
2040 \ZREF@xr@found\space
2041 \ifZREF@xr@zreflabel zref\else LaTeX\fi\space
2042 label(s) found%
2043 \ifnum\ZREF@xr@ignored@empty>0 %
2044 ,\MessageBreak
2045 \ZREF@xr@ignored@empty\space empty label(s) ignored%
2046 \fi
2047 \ifnum\ZREF@xr@ignored@zref>0 %
2048 ,\MessageBreak
2049 \ZREF@xr@ignored@zref\space
2050 duplicated zref label(s) ignored%
2051 \fi
2052 \ifnum\ZREF@xr@ignored@ltx>0 %
2053 ,\MessageBreak
2054 \ZREF@xr@ignored@ltx\space
2055 duplicated latex label(s) ignored%
2056 \fi
2057 }%
2058 \endgroup
2059 \fi
2060 \ifx\ZREF@xr@filelist\ltx@empty
2061 \else
2062 \edef\ZREF@xr@file{%
2063 \expandafter\ltx@car\ZREF@xr@filelist\@nil
2064 }%
2065 \edef\ZREF@xr@filelist{%
2066 \expandafter\ltx@cdr\ZREF@xr@filelist\ltx@empty\@nil
2067 }%
2068 \expandafter\ZREF@xr@checkfile
2069 \fi
2070 }%

```

\ZREF@xr@processfile

```

2071 \def\ZREF@xr@processfile{%

```

```

2072 \read\@inputcheck to\ZREF@xr@line
2073 \expandafter\ZREF@xr@processline\ZREF@xr@line..\ZREF@nil
2074 \ifeof\@inputcheck
2075 \else
2076   \expandafter\ZREF@xr@processfile
2077 \fi
2078 }%

```

\ZREF@xr@processline The most work must be done for analyzing the arguments of \newlabel.

```

2079 \long\def\ZREF@xr@processline#1#2#3\ZREF@nil{%
2080   \def\x{#1}%
2081   \toks@{#2}%
2082   \ifZREF@xr@zreflabel
2083     \ifx\x\ZREF@xr@zref@newlabel
2084       \expandafter
2085         \ZREF@xr@process@zreflabel\ZREF@xr@line..\ZREF@nil
2086       \fi
2087     \else
2088       \ifx\x\ZREF@xr@newlabel
2089         \expandafter
2090           \ZREF@xr@process@label\ZREF@xr@line... []\ZREF@nil
2091         \fi
2092       \fi
2093     \ifx\x\ZREF@xr@@input
2094       \edef\ZREF@xr@filelist{%
2095         \tex@unexpanded\expandafter{\ZREF@xr@filelist}%
2096         {\filename@area\the\toks@}}%
2097       }%
2098     \fi
2099 }%
2100 \def\ZREF@xr@process@zreflabel\zref@newlabel#1#2#3\ZREF@nil{%
2101   \edef\ZREF@xr@refname{Z@R@ZREF@xr@prefix#1}%
2102   \edef\ZREF@xr@found{\the\numexpr\ZREF@xr@found+1\relax}%
2103   \def\x{#2}%
2104   \edef\ZREF@xr@tempname{$temp$}%
2105   \edef\ZREF@xr@temprefname{Z@R@ZREF@xr@tempname}%
2106   \let\ZREF@xr@list\x
2107   \ifx\ZREF@xr@list\ltx@empty
2108     \PackageWarningNoLine{zref-xr}{%
2109       Label ‘#1’ without properties ignored\MessageBreak
2110       in file ‘\ZREF@xr@file’%
2111     }%
2112     \edef\ZREF@xr@ignored@empty{%
2113       \the\numexpr\ZREF@xr@ignored@empty+1\relax
2114     }%
2115   \else
2116     \expandafter\ZREF@xr@checklist\x\ZREF@nil
2117     \expandafter\let\csname\ZREF@xr@temprefname\endcsname\x
2118     \expandafter\ltx@LocalAppendToMacro
2119     \csname\ZREF@xr@temprefname\endcsname\expandafter\endcsname
2120     \expandafter{%
2121       \expandafter\externaldocument\expandafter{%
2122         \ZREF@xr@externalfile
2123       }%
2124     }%
2125     \ZREF@xr@urlcheck\ZREF@xr@tempname
2126     \ifZREF@xr@tozreflabel
2127       \ifundefined{\ZREF@xr@refname}{%
2128         \ifZREF@xr@verbose
2129           \PackageInfo{zref-xr}{%
2130             Import to zref label ‘\ZREF@xr@tempname#1’%
2131           }%
2132         \fi

```

```

2133     \global\expandafter
2134     \let\csname\ZREF@xr@refname\expandafter\endcsname
2135     \csname\ZREF@xr@temprefname\endcsname
2136   }{%
2137     \ZREF@xr@zref@ignorewarning{\ZREF@xr@prefix#1}%
2138   }%
2139   \fi
2140   \ifZREF@xr@toltxlabel
2141     \ZREF@xr@tolabel{\ZREF@xr@tempname}{\ZREF@xr@prefix#1}%
2142   \fi
2143 \fi
2144 }%
2145 \def\ZREF@xr@process@label\newlabel#1#2#3[#4]#5\ZREF@nil{%
2146 \def\ZREF@xr@refname{Z@R@\ZREF@xr@prefix#1}%
2147 \edef\ZREF@xr@found{\the\numexpr\ZREF@xr@found+1\relax}%
2148 \def\x{#2}%
2149 \edef\ZREF@xr@tempname{$temp$}%
2150 \edef\ZREF@xr@temprefname{Z@R@\ZREF@xr@tempname}%
2151 \expandafter\ZREF@xr@scanparams
2152     \csname\ZREF@xr@temprefname\expandafter\endcsname
2153     \x{}{}{}{}\ZREF@nil
2154 \ifx\#4\%
2155 \else
2156   % ntheorem knows an optional argument at the end of \newlabel
2157   \ZREF@NewPropTheotype
2158   \expandafter\ltx@LocalAppendToMacro
2159     \csname\ZREF@xr@temprefname\endcsname{\theotype{#4}}%
2160 \fi
2161 \expandafter\ltx@LocalAppendToMacro
2162 \csname\ZREF@xr@temprefname\expandafter\endcsname\expandafter{%
2163   \expandafter\externaldocument\expandafter{%
2164     \ZREF@xr@externalfile
2165   }%
2166 }%
2167 \ZREF@xr@urlcheck\ZREF@xr@tempname
2168 \ifZREF@xr@tozreflabel
2169   \@ifundefined{\ZREF@xr@refname}{%
2170     \ifZREF@xr@verbose
2171       \PackageInfo{zref-xr}{%
2172         Import to zref label '\ZREF@xr@prefix#1'%
2173       }%
2174     \fi
2175     \global\expandafter
2176     \let\csname\ZREF@xr@refname\expandafter\endcsname
2177     \csname\ZREF@xr@temprefname\endcsname
2178   }{%
2179     \ZREF@xr@zref@ignorewarning{\ZREF@xr@prefix#1}%
2180   }%
2181 \fi
2182 \ifZREF@xr@toltxlabel
2183   \ZREF@xr@tolabel{\ZREF@xr@tempname}{\ZREF@xr@prefix#1}%
2184 \fi
2185 }
2186 \def\ZREF@xr@zref@newlabel{\zref@newlabel}%
2187 \def\ZREF@xr@newlabel{\newlabel}%
2188 \def\ZREF@xr@@input{\@input}%
2189 \def\ZREF@xr@relax{\relax}%

\ZREF@xr@tolabel

2190 \def\ZREF@xr@tolabel#1#2{%
2191   \ifZREF@xr@verbose
2192     \PackageInfo{zref-xr}{%
2193       Import to LaTeX label '#2'%

```

```

2194 }%
2195 \fi
2196 \zref@wrapper@unexpanded{%
2197   \expandafter\xdef\csname r@#2\endcsname{%
2198     {%
2199       \ltx@ifundefined{M@TitleReference}{%
2200         \ltx@ifundefined{TR@TitleReference}{%
2201           \zref@extractdefault{#1}{default}{}%
2202         }{%
2203           \noexpand\TR@TitleReference
2204             {\zref@extractdefault{#1}{default}{}}}%
2205           {\zref@extractdefault{#1}{title}{}}}%
2206         }%
2207       }{%
2208         \noexpand\M@TitleReference
2209           {\zref@extractdefault{#1}{default}{}}}%
2210           {\zref@extractdefault{#1}{title}{}}}%
2211       }%
2212     }%
2213     {\zref@extractdefault{#1}{page}{}}}%
2214   \ltx@ifpackageloaded{nameref}{%
2215     {\zref@extractdefault{#1}{title}{}}}%
2216     {\zref@extractdefault{#1}{anchor}{}}}%
2217     \zref@ifrefcontainsprop{#1}{urluse}{%
2218       {\zref@extractdefault{#1}{urluse}{}}}%
2219     }{%
2220       {\zref@extractdefault{#1}{url}{}}}%
2221     }%
2222   }{%
2223   }%
2224 }%
2225 }

```

`\ZREF@xr@zref@ignorewarning`

```

2226 \def\ZREF@xr@zref@ignorewarning#1{%
2227   \PackageWarningNoLine{zref-xr}{%
2228     Zref label ‘#1’ is already in use\MessageBreak
2229     in file ‘\ZREF@xr@file’%
2230   }%
2231   \edef\ZREF@xr@ignored@zref{%
2232     \the\numexpr\ZREF@xr@ignored@zref+1%
2233   }%
2234 }%

```

`\ZREF@xr@ltx@ignorewarning`

```

2235 \def\ZREF@xr@ltx@ignorewarning#1{%
2236   \PackageWarningNoLine{zref-xr}{%
2237     LaTeX label ‘#1’ is already in use\MessageBreak
2238     in file ‘\ZREF@xr@file’%
2239   }%
2240   \edef\ZREF@xr@ignored@ltx{%
2241     \the\numexpr\ZREF@xr@ignored@ltx+1%
2242   }%
2243 }%

```

`\ZREF@xr@checklist`

```

2244 \def\ZREF@xr@checklist#1#2#3\ZREF@nil{%
2245   \ifx\@undefined#1\relax
2246     \expandafter\ZREF@xr@checkkey\string#1\@nil
2247     \fi
2248     \ifx\#3\%
2249     \else

```



```

2250 \ltx@ReturnAfterFi{%
2251 \ZREF@xr@checklist#3\ZREF@nil
2252 }%
2253 \fi
2254 }%
2255 \def\ZREF@xr@checkkey#1#2\@nil{%
2256 \zref@ifpropundefined{#2}{%
2257 \zref@newprop{#2}{}%
2258 }{%
2259 }%

```

#### \ZREF@xr@scanparams

```

2260 \def\ZREF@xr@scanparams#1#2#3#4#5#6#7\ZREF@nil{%
2261 \let#1\ltx@empty
2262 \ZREF@foundfalse
2263 \ZREF@xr@scantitleref#1#2\TR@TitleReference{}{}\ZREF@nil
2264 \ifZREF@found
2265 \else
2266 \ltx@LocalAppendToMacro#1{\default{#2}}%
2267 \fi
2268 % page
2269 \ltx@LocalAppendToMacro#1{\page{#3}}%
2270 % nameref title
2271 \ifZREF@found
2272 \else
2273 \ifx\\#4\\%
2274 \else
2275 \def\ZREF@xr@temp{#4}%
2276 \ifx\ZREF@xr@temp\ZREF@xr@relax
2277 \else
2278 \ltx@LocalAppendToMacro#1{\title{#4}}%
2279 \fi
2280 \fi
2281 \fi
2282 % anchor
2283 \ifx\\#5\\%
2284 \else
2285 \ltx@LocalAppendToMacro#1{\anchor{#5}}%
2286 \fi
2287 \ifx\\#6\\%
2288 \else
2289 \ifZREF@xr@urluse
2290 \ZREF@xr@AddURL{#6}%
2291 \expandafter\ltx@LocalAppendToMacro\expandafter#1%
2292 \expandafter{%
2293 \expandafter\urluse\expandafter{%
2294 \csname Z@U@\ZREF@xr@theURL\endcsname
2295 }%
2296 }%
2297 \else
2298 \ltx@LocalAppendToMacro#1{\url{#6}}%
2299 \fi
2300 \fi
2301 }%

```

#### \ZREF@xr@scantitleref

```

2302 \def\ZREF@xr@scantitleref#1#2\TR@TitleReference#3#4#5\ZREF@nil{%
2303 \ifx\\#5\\%
2304 \else
2305 \ltx@LocalAppendToMacro#1{%
2306 \default{#3}%
2307 \title{#4}%

```

```

2308 }%
2309 \ZREF@foundtrue
2310 \fi
2311 }%

```

`\ZREF@xr@urlcheck`

```

2312 \def\ZREF@xr@urlcheck#1{%
2313 \zref@ifrefcontainsprop{#1}{anchor}{%
2314 \zref@ifrefcontainsprop{#1}{url}{%
2315 }{%
2316 \expandafter
2317 \ltx@LocalAppendToMacro\csname Z@R@#1\expandafter\endcsname
2318 \expandafter{%
2319 \csname url\ifZREF@xr@urluse use\fi
2320 \expandafter\endcsname\expandafter{\ZREF@xr@url}%
2321 }%
2322 }%
2323 }{%
2324 }%
2325 }%

2326 </xr>

```

## 6.17 Module hyperref

UNFINISHED :-(

```

2327 <*hyperref>
2328 \NeedsTeXFormat{LaTeX2e}
2329 \ProvidesPackage{zref-hyperref}%
2330 [2011/03/18 v2.21 Module hyperref for zref (HO)]%
2331 \RequirePackage{zref-base}[2011/03/18]
2332 \ifx\ZREF@base@ok Y%
2333 \else
2334 \expandafter\endinput
2335 \fi

2336 \ZREF@NewPropAnchor
2337 \zref@addprop\ZREF@mainlist{anchor}%
2338 </hyperref>

```

## 6.18 Module savepos

Module savepos provides an interface for pdfTeX's `\pdfsavepos`, see the manual for pdfTeX.

### 6.18.1 Identification

```

2339 <*savepos>
2340 \NeedsTeXFormat{LaTeX2e}
2341 \ProvidesPackage{zref-savepos}%
2342 [2011/03/18 v2.21 Module savepos for zref (HO)]%
2343 \RequirePackage{zref-base}[2011/03/18]
2344 \ifx\ZREF@base@ok Y%
2345 \else
2346 \expandafter\endinput
2347 \fi

```

### 6.18.2 Availability

First we check, whether the feature is available.

```

2348 \ltx@ifundefined{pdfsavepos}{%
2349 \PackageError\ZREF@name{%

```

```

2350   \string\pdfsavepos\space is not supported.\MessageBreak
2351   It is provided by pdfTeX (1.40) or XeTeX%
2352 } \ZREF@UpdatePdfTeX
2353 \endinput
2354 }{}%
```

In PDF mode we are done. However support for DVI mode was added later in version 1.40.0. In earlier versions `\pdfsavepos` is defined, but its execution raises an error. Note that  $\text{\LaTeX}$  also provides `\pdfsavepos`.

```

2355 \RequirePackage{ifpdf}
2356 \ifpdf
2357 \else
2358   \ltx@ifundefined{pdftexversion}{%
2359   }{%
2360     \ifnum\pdftexversion<140 %
2361       \PackageError\ZREF@name{%
2362         \string\pdfsavepos\space is not supported in DVI mode%
2363         \MessageBreak
2364         of this pdfTeX version%
2365       } \ZREF@UpdatePdfTeX
2366       \expandafter\expandafter\expandafter\endinput
2367     \fi
2368   }%
2369 \fi
```

### 6.18.3 Setup

```

2370 \zref@newlist{savepos}
2371 \zref@newprop*{posx}[0]{\the\pdflastxpos}
2372 \zref@newprop*{posy}[0]{\the\pdflastypos}
2373 \zref@addprops{savepos}{posx, posy}
```

### 6.18.4 User macros

`\zref@savepos`

```

2374 \def\zref@savepos{%
2375   \if@filesw
2376     \pdfsavepos
2377   \fi
2378 }
```

`\zsavapos` The current location is stored in a reference with the given name.

```

2379 \ZREF@ifdefinable\zsavapos\def{%
2380   #1{%
2381     \@bsphack
2382     \if@filesw
2383       \zref@savepos
2384       \zref@labelbylist{#1}{savepos}%
2385     \fi
2386     \@esphack
2387   }%
2388 }
```

`\zposx` The horizontal and vertical position are available by `\zposx` and `\zposy`. Do not rely on absolute positions. They differ in DVI and PDF mode of  $\text{\LaTeX}$ . Use differences instead. The unit of the position numbers is sp.

`\zposy`

```

2389 \newcommand*{\zposx}[1]{%
2390   \zref@extract{#1}{posx}%
2391 }%
2392 \newcommand*{\zposy}[1]{%
2393   \zref@extract{#1}{posy}%
2394 }%
```

Typically horizontal and vertical positions are used inside calculations. Therefore the extracting macros should be expandable and babel's patch is not applicable.

Also it is in the responsibility of the user to marked used positions by `\zrefused` in order to notify L<sup>A</sup>T<sub>E</sub>X about undefined references.

`\ZREF@savepos@ok`

```
2395 \let\ZREF@savepos@ok=Y
```

```
2396 </savepos>
```

## 6.19 Module `abspos`

### 6.19.1 Identification

```
2397 <*abspos>
```

```
2398 \NeedsTeXFormat{LaTeX2e}
```

```
2399 \ProvidesPackage{zref-abspos}%
```

```
2400 [2011/03/18 v2.21 Module abspos for zref (HO)]%
```

```
2401 \RequirePackage{zref-base}[2011/03/18]
```

```
2402 \ifx\ZREF@base@ok Y%
```

```
2403 \else
```

```
2404 \expandafter\endinput
```

```
2405 \fi
```

```
2406 \RequirePackage{zref-savepos}[2011/03/18]
```

```
2407 \ifx\ZREF@savepos@ok Y%
```

```
2408 \else
```

```
2409 \expandafter\endinput
```

```
2410 \fi
```

```
2411 \RequirePackage{zref-pagelayout}[2011/03/18]
```

```
2412 \zref@addprop{savepos}{abspage}
```

```
2413 </abspos>
```

## 6.20 Module `dotfill`

```
2414 <*dotfill>
```

```
2415 \NeedsTeXFormat{LaTeX2e}
```

```
2416 \ProvidesPackage{zref-dotfill}%
```

```
2417 [2011/03/18 v2.21 Module dotfill for zref (HO)]%
```

```
2418 \RequirePackage{zref-base}[2011/03/18]
```

```
2419 \ifx\ZREF@base@ok Y%
```

```
2420 \else
```

```
2421 \expandafter\endinput
```

```
2422 \fi
```

For measuring the width of `\zdotfill` we use the features provided by module `savepos`.

```
2423 \RequirePackage{zref-savepos}[2011/03/18]
```

For automatically generated label names we use the unique counter of module `base`.

```
2424 \zref@require@unique
```

Configuration is done by the key value interface of package `keyval`.

```
2425 \RequirePackage{keyval}
```

The definitions of the keys follow.

```
2426 \define@key{ZREF@DF}{unit}{%
```

```
2427 \def\ZREF@df@unit{#1}%
```

```
2428 }
```

```
2429 \define@key{ZREF@DF}{min}{%
```

```
2430 \def\ZREF@df@min{#1}%
```

```
2431 }
```

```
2432 \define@key{ZREF@DF}{dot}{%
```

```

2433 \def\ZREF@df@dot{#1}%
2434 }

Defaults are set, see user interface.
2435 \providecommand\ZREF@df@min{2}
2436 \providecommand\ZREF@df@unit{.44em}
2437 \providecommand\ZREF@df@dot{.}

\zdotfillsetup Configuration of \zdotfill is done by \zdotfillsetup.
2438 \newcommand*{\zdotfillsetup}{\kvsetkeys{ZREF@DF}}

\zdotfill \zdotfill sets labels at the left and the right to get the horizontal position.
\zsavepos is not used, because we do not need the vertical position.
2439 \ZREF@IfDefinable\zdotfill\def{%
2440   {%
2441     \leavevmode
2442     \global\advance\c@zref@unique\ltx@one
2443     \begingroup
2444     \def\ZREF@temp{zref@\number\c@zref@unique}%
2445     \pdfsavepos
2446     \zref@labelbyprops{\thezref@unique L}{posx}%
2447     \setlength{\dimen@}{\ZREF@df@unit}%
2448     \zref@ifrefundefined{\thezref@unique R}{%
2449       \ZREF@dotfill
2450     }{%
2451       \ifnum\numexpr\zposx{\thezref@unique R}%
2452         -\zposx{\thezref@unique L}\relax
2453       <\dimexpr\ZREF@df@min\dimen@\relax
2454       \hfill
2455     \else
2456       \ZREF@dotfill
2457     \fi
2458   }%
2459   \pdfsavepos
2460   \zref@labelbyprops{\thezref@unique R}{posx}%
2461 \endgroup
2462 \kern\z@
2463 }%
2464 }

\ZREF@dotfill Help macro that actually sets the dots.
2465 \def\ZREF@dotfill{%
2466 \cleaders\hb@xt@\dimen@{\hss\ZREF@df@dot\hss}\hfill
2467 }

2468 </dotfill)

6.21 Module env
2469 <*env)
2470 \NeedsTeXFormat{LaTeX2e}
2471 \ProvidesPackage{zref-env}%
2472 [2011/03/18 v2.21 Module env for zref (HO)]%
2473 \RequirePackage{zref-base}[2011/03/18]
2474 \ifx\ZREF@base@ok Y%
2475 \else
2476 \expandafter\endinput
2477 \fi

2478 \zref@newprop{envname}[]{\@currenvir}
2479 \zref@newprop{envline}[]{\zref@env@line}

\zref@env@line Macro \zref@env@line extracts the line number from \@currenvline.
2480 \def\zref@env@line{%

```

```

2481 \ifx\@currentvline\ltx@empty
2482 \else
2483 \expandafter
2484 \ZREF@ENV@line\@currentvline\ltx@empty line \ltx@empty\@nil
2485 \fi
2486 }

```

\ZREF@ENV@line

```

2487 \def\ZREF@ENV@line#1line #2\ltx@empty#3\@nil{#2}%
2488 </env>

```

## 7 Test

### 7.1 \zref@localaddprop

```

2489 <*test1>
2490 \NeedsTeXFormat{LaTeX2e}
2491 \nofiles
2492 \documentclass{article}
2493 \usepackage{zref-base}[2011/03/18]
2494 \usepackage{qstest}
2495 \IncludeTests{*}
2496 \LogTests{log}{*}{*}
2497
2498 \makeatletter
2499 \def\ExpectList#1#2{%
2500 \expandafter\expandafter\expandafter\Expect
2501 \expandafter\expandafter\expandafter{\csname Z@L@#1\endcsname}{#2}%
2502 }
2503 \begin{qstest}{localaddprop}{localaddprop}
2504 \ExpectList{main}{\default\page}%
2505 \Expect{undefined}*{\meaning\foobar}%
2506 \zref@newprop{foobar}{FOO}%
2507 \Expect{undefined}*{\meaning\foobar}%
2508 \zref@newlist{alist}%
2509 \ExpectList{alist}{}%
2510 \begingroup
2511 \zref@localaddprop{main}{foobar}%
2512 \Expect{undefined}*{\meaning\foobar}%
2513 \ExpectList{main}{\default\page\foobar}%
2514 \zref@localaddprop{alist}{page}%
2515 \ExpectList{alist}{\page}%
2516 \endgroup
2517 \ExpectList{main}{\default\page}%
2518 \ExpectList{alist}{}%
2519 \zref@addprop{alist}{foobar}%
2520 \ExpectList{alist}{\foobar}%
2521 \Expect{undefined}*{\meaning\foobar}%
2522 \end{qstest}
2523 \@@end
2524 </test1>

```

### 7.2 Module base

```

2525 <*test-base>
2526 \NeedsTeXFormat{LaTeX2e}
2527 \documentclass{article}
2528 \usepackage{zref-base,zref-titleref}[2011/03/18]
2529 \usepackage{qstest}
2530 \IncludeTests{*}
2531 \LogTests{log}{*}{*}
2532

```

```

2533 \makeatletter
2534 \newcommand*{\DefExpand}[2]{%
2535   \expandafter\expandafter\expandafter\def
2536   \expandafter\expandafter\expandafter#1%
2537   \expandafter\expandafter\expandafter{#2}%
2538   \@onelevel@sanitize#1%
2539 }
2540 \newcommand*{\Test}[3]{%
2541   \Expect{#2}*{#1}%
2542   \zref@wrapper@unexpanded{%
2543     \Expect*{#3}*{#1}%
2544   }%
2545   \DefExpand\x{#1}%
2546   \Expect*{#3}*{\x}%
2547 }
2548 \makeatother
2549
2550 \begin{document}
2551 \section{\textit{Hello} \textbf{World}}
2552 \label{sec:hello}
2553 \makeatletter
2554 \zref@newprop{foo}[\@empty D\@empty efault]{\@empty V\@empty alue}
2555 \begin{qstest}{getcurrent}{getcurrent}
2556   \Test{\zref@getcurrent{foo}}%
2557     {Value}{\noexpand\@empty V\noexpand\@empty alue}%
2558   \Test{\zref@getcurrent{xy}}{-}{-}%
2559 \end{qstest}
2560 \begin{qstest}{extract}{extract}
2561   \def\textbf#1{<#1>}%
2562   \def\textit#1{[#1]}% hash-ok
2563   \Test{\zref@extractdefault{xy}{page}{\@empty D\@empty efault}}%
2564     {Default}{\noexpand\@empty D\noexpand\@empty efault}%
2565   \Test{\zref@extractdefault{sec:hello}{foo}{\@empty A\@empty B}}%
2566     {AB}{\noexpand\@empty A\noexpand\@empty B}%
2567   \Test{\zref@extract{sec:hello}{foo}}%
2568     {Default}{\noexpand\@empty D\noexpand\@empty efault}%
2569   \zref@ifrefundefined{sec:hello}{%
2570 }{%
2571   \Test{\zref@extract{sec:hello}{default}}{1}{1}%
2572   \Test{\zref@extract{sec:hello}{title}}%
2573     {[Hello] <World>}%
2574     {\noexpand\textit{Hello} \noexpand\textbf{World}}%
2575 }%
2576 \end{qstest}
2577 \end{document}
2578 </test-base>

```

### 7.3 Module runs

```

2579 <*test-runs>
2580 \NeedsTeXFormat{LaTeX2e}
2581 \documentclass{article}
2582 \usepackage{zref-runs}[2011/03/18]
2583 \usepackage{qstest}
2584 \IncludeTests{*}
2585 \LogTests{log}{*}{*}
2586
2587 \begin{qstest}{zruns-preamble}{zruns-preamble}
2588   \Expect{0}*{\zruns}%
2589 \end{qstest}
2590
2591 \AtBeginDocument{%
2592   \begin{qstest}{zruns-atbegindocument}{zruns-atbegindocument}%

```

```

2593 \Expect*{\number\ExpectRuns}*{\zruns}%
2594 \end{qstest}%
2595 }
2596
2597 \begin{document}
2598 \begin{qstest}{zruns-document}{zruns-document}
2599 \Expect*{\number\ExpectRuns}*{\zruns}%
2600 \end{qstest}
2601 \end{document}
2602 </test-runs>

```

## 7.4 Module titleref

```

2603 <*test-titleref-memoir>
2604 \NeedsTeXFormat{LaTeX2e}
2605 \documentclass{memoir}
2606 \usepackage{zref-titleref}[2011/03/18]
2607 \usepackage{qstest}
2608 \IncludeTests{*}
2609 \LogTests{log}{*}{*}
2610 \begin{document}
2611 \makeatletter
2612 \def\List{}
2613 \def\Label#1{%
2614 \zref@label{#1}%
2615 \g@addto@macro\List{%
2616 \par
2617 #1: [\ztitleref{#1}]%
2618 }%
2619 \mbox{}}%
2620 \zref@refused{#1}%
2621 \zref@ifrefundefined{#1}{%
2622 }{%
2623 \begingroup
2624 \edef\x{\zref@extract{#1}{title}}%
2625 \Expect{OK/}*{\expandafter\ltx@carthree\x}{-}\@nil}%
2626 \endgroup
2627 }%
2628 }
2629 \def\Test#1{%
2630 \csname#1\endcsname*{OK/#1}%
2631 \Label{#1*}%
2632 \csname#1\endcsname{OK/#1}%
2633 \Label{#1}%
2634 \csname#1\endcsname[OK/#1-toc]%
2635 {WRONG-in-titleref/#1-toc-2}%
2636 \Label{#1-toc}%
2637 \expandafter\ifx\csname#1\endcsname\part
2638 \else
2639 \headnamereffalse
2640 \csname#1\endcsname[OK/#1-th-toc]%
2641 [WRONG-in-titleref/#1-th-toc-2]%
2642 {WRONG-in-titleref/#1-th-toc-3}%
2643 \Label{#1-th-toc}%
2644 \headnamereftrue
2645 \csname#1\endcsname[WRONG-in-titleref/#1-th-head-1]%
2646 [OK/#1-th-head]%
2647 {WRONG-in-titleref/#1-th-head-3}%
2648 \Label{#1-th-head}%
2649 \fi
2650 }
2651 \begin{qstest}{section}{section}
2652 \@for\x:=part,chapter,section,subsection,subsubsection\do{%

```



```

2653     \expandafter\Test\expandafter{\x}%
2654   }%
2655 \end{qstest}
2656 \newpage
2657 \List
2658 \end{document}
2659 </test-titleref-memoir>

```

## 8 Installation

### 8.1 Download

**Package.** This package is available on CTAN<sup>2</sup>:

[CTAN:macros/latex/contrib/oberdiek/zref.dtx](#) The source file.

[CTAN:macros/latex/contrib/oberdiek/zref.pdf](#) Documentation.

**Bundle.** All the packages of the bundle ‘oberdiek’ are also available in a TDS compliant ZIP archive. There the packages are already unpacked and the documentation files are generated. The files and directories obey the TDS standard.

[CTAN:install/macros/latex/contrib/oberdiek.tds.zip](#)

*TDS* refers to the standard “A Directory Structure for  $\TeX$  Files” ([CTAN:tds/tds.pdf](#)). Directories with `texmf` in their name are usually organized this way.

### 8.2 Bundle installation

**Unpacking.** Unpack the `oberdiek.tds.zip` in the TDS tree (also known as `texmf` tree) of your choice. Example (linux):

```
unzip oberdiek.tds.zip -d ~/texmf
```

**Script installation.** Check the directory `TDS:scripts/oberdiek/` for scripts that need further installation steps. Package `attachfile2` comes with the Perl script `pdfatfi.pl` that should be installed in such a way that it can be called as `pdfatfi`. Example (linux):

```
chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/
```

### 8.3 Package installation

**Unpacking.** The `.dtx` file is a self-extracting `docstrip` archive. The files are extracted by running the `.dtx` through plain  $\TeX$ :

```
tex zref.dtx
```

**TDS.** Now the different files must be moved into the different directories in your installation TDS tree (also known as `texmf` tree):

<sup>2</sup><ftp://ftp.ctan.org/tex-archive/>

<code>zref.sty</code>	<code>→ tex/latex/oberdiek/zref.sty</code>
<code>zref-base.sty</code>	<code>→ tex/latex/oberdiek/zref-base.sty</code>
<code>zref-abspage.sty</code>	<code>→ tex/latex/oberdiek/zref-abspage.sty</code>
<code>zref-abspos.sty</code>	<code>→ tex/latex/oberdiek/zref-abspos.sty</code>
<code>zref-counter.sty</code>	<code>→ tex/latex/oberdiek/zref-counter.sty</code>
<code>zref-dotfill.sty</code>	<code>→ tex/latex/oberdiek/zref-dotfill.sty</code>
<code>zref-env.sty</code>	<code>→ tex/latex/oberdiek/zref-env.sty</code>
<code>zref-hyperref.sty</code>	<code>→ tex/latex/oberdiek/zref-hyperref.sty</code>
<code>zref-lastpage.sty</code>	<code>→ tex/latex/oberdiek/zref-lastpage.sty</code>
<code>zref-marks.sty</code>	<code>→ tex/latex/oberdiek/zref-marks.sty</code>
<code>zref-nextpage.sty</code>	<code>→ tex/latex/oberdiek/zref-nextpage.sty</code>
<code>zref-pageattr.sty</code>	<code>→ tex/latex/oberdiek/zref-pageattr.sty</code>
<code>zref-pagelayout.sty</code>	<code>→ tex/latex/oberdiek/zref-pagelayout.sty</code>
<code>zref-perpage.sty</code>	<code>→ tex/latex/oberdiek/zref-perpage.sty</code>
<code>zref-runs.sty</code>	<code>→ tex/latex/oberdiek/zref-runs.sty</code>
<code>zref-savepos.sty</code>	<code>→ tex/latex/oberdiek/zref-savepos.sty</code>
<code>zref-thepage.sty</code>	<code>→ tex/latex/oberdiek/zref-thepage.sty</code>
<code>zref-titleref.sty</code>	<code>→ tex/latex/oberdiek/zref-titleref.sty</code>
<code>zref-totpages.sty</code>	<code>→ tex/latex/oberdiek/zref-totpages.sty</code>
<code>zref-user.sty</code>	<code>→ tex/latex/oberdiek/zref-user.sty</code>
<code>zref-xr.sty</code>	<code>→ tex/latex/oberdiek/zref-xr.sty</code>
<code>zref.pdf</code>	<code>→ doc/latex/oberdiek/zref.pdf</code>
<code>zref-example.tex</code>	<code>→ doc/latex/oberdiek/zref-example.tex</code>
<code>zref-example-lastpage.tex</code>	<code>→ doc/latex/oberdiek/zref-example-lastpage.tex</code>
<code>zref-example-nextpage.tex</code>	<code>→ doc/latex/oberdiek/zref-example-nextpage.tex</code>
<code>test/zref-test1.tex</code>	<code>→ doc/latex/oberdiek/test/zref-test1.tex</code>
<code>test/zref-test-base.tex</code>	<code>→ doc/latex/oberdiek/test/zref-test-base.tex</code>
<code>test/zref-test-runs.tex</code>	<code>→ doc/latex/oberdiek/test/zref-test-runs.tex</code>
<code>test/zref-test-titleref-memoir.tex</code>	<code>→ doc/latex/oberdiek/test/zref-test-titleref-memoir.tex</code>
<code>zref.dtx</code>	<code>→ source/latex/oberdiek/zref.dtx</code>

If you have a `docstrip.cfg` that configures and enables `docstrip`'s TDS installing feature, then some files can already be in the right place, see the documentation of `docstrip`.

## 8.4 Refresh file name databases

If your  $\text{T}_{\text{E}}\text{X}$  distribution (`te $\text{T}_{\text{E}}\text{X}$` , `mik $\text{T}_{\text{E}}\text{X}$` , ...) relies on file name databases, you must refresh these. For example, `te $\text{T}_{\text{E}}\text{X}$`  users run `texhash` or `mktexlsr`.

## 8.5 Some details for the interested

**Attached source.** The PDF documentation on CTAN also includes the `.dtx` source file. It can be extracted by AcrobatReader 6 or higher. Another option is `pdftk`, e.g. unpack the file into the current directory:

```
pdftk zref.pdf unpack_files output .
```

**Unpacking with  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ .** The `.dtx` chooses its action depending on the format:

**plain  $\text{T}_{\text{E}}\text{X}$ :** Run `docstrip` and extract the files.

**$\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ :** Generate the documentation.

If you insist on using  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  for `docstrip` (really, `docstrip` does not need  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{zref.dtx}
```

Do not forget to quote the argument according to the demands of your shell.

**Generating the documentation.** You can use both the `.dtx` or the `.drv` to generate the documentation. The process can be configured by the configuration file `ltxdoc.cfg`. For instance, put this line into this file, if you want to have A4 as paper format:

```
\PassOptionsToClass{a4paper}{article}
```

An example follows how to generate the documentation with `pdf $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$` :

```
pdflatex zref.dtx
makeindex -s gind.ist zref.idx
pdflatex zref.dtx
makeindex -s gind.ist zref.idx
pdflatex zref.dtx
```

## 9 References

- [1] Package footmisc, Robin Fairbairns, 2004/01/23 v5.3a.[CTAN:macros/latex/contrib/footmisc/footmisc.dtx](#)
- [2] Package hyperref, Sebastian Rahtz, Heiko Oberdiek, 2006/08/16 v6.75c.[CTAN:macros/latex/contrib/hyperref/](#)
- [3] Package lastpage, Jeff Goldberg, 1994/06/25 v0.1b.[CTAN:macros/latex/contrib/lastpage/](#)
- [4] Package nameref, Sebastian Rahtz, Heiko Oberdiek, 2006/02/12 v2.24.[CTAN:macros/latex/contrib/hyperref/nameref.dtx](#)
- [5] Package perpage, David Kastrup, 2002/12/20 v1.0.[CTAN:macros/latex/contrib/bigfoot/perpage.dtx](#)
- [6] Package titleref, Donald Arsenaу, 2001/04/05 v3.1.[CTAN:macros/latex/contrib/misc/titleref.sty](#)
- [7] Package totpages, Wilhelm Müller, 1999/07/14 v1.00.[CTAN:macros/latex/contrib/totpages/](#)
- [8] Package xr, David Carlisle, 1994/05/28 v5.02.[CTAN:macros/latex/required/tools/xr.pdf](#)
- [9] Package xr-hyper, David Carlisle, 2000/03/22 v6.00beta4.[CTAN:macros/latex/contrib/hyperref/xr-hyper.sty](#)

## 10 History

### [2006/02/20 v1.0]

- First version.

### [2006/05/03 v1.1]

- Module perpage added.
- Module redesign as packages.

### [2006/05/25 v1.2]

- Module dotfillmin added.
- Module base: macros `\zref@require@unique` and `\thezref@unique` added (used by modules `titleref` and `dotfillmin`).

### [2006/09/08 v1.3]

- Typo fixes and English cleanup by Per Starback.

### [2007/01/23 v1.4]

- Typo in macro name fixed in documentation.

**[2007/02/18 v1.5]**

- `\zref@getcurrent` added (suggestion of Igor Akkerman).
- Module `savepos` also supports  $\LaTeX$ .

**[2007/04/06 v1.6]**

- Fix in modules `abspage` and `base`: Now counter `abspage` and `zref@unique` are not remembered by `\include`.
- Beamer support for module `titleref`.

**[2007/04/17 v1.7]**

- Package `atbegshi` replaces `everyshi`.

**[2007/04/22 v1.8]**

- `\zref@wrapper@babel` and `\zref@refused` are now expandable if `babel` is not used or `\if@safe@actives` is already set to true. (Feature request of Josselin Noirel)

**[2007/05/02 v1.9]**

- Module `titleref`: Some support for `\caption` of package `longtable`, but only if `\label` is given after `\caption`.

**[2007/05/06 v2.0]**

- Uses package `etexcmds` for accessing  $\epsilon$ - $\TeX$ 's `\unexpanded`.

**[2007/05/28 v2.1]**

- Module `titleref` supports caption of package `listings`.
- Fixes in module `titleref` for support of packages `titlesec` and `longtable`.

**[2008/09/21 v2.2]**

- Module `base`: `\zref@iflistcontainsprop` is documented, but a broken `\zref@listcontainsprop` implemented. Name and implementation fixed (thanks Ohad Kammar).

**[2008/10/01 v2.3]**

- `\zref@localaddprop` added (feature request of Ohad Kammar).
- Module `lastpage`: list 'LastPage' added. Label 'LastPage' will use the properties of this list (default is empty) along with the properties of the main list.

**[2009/08/07 v2.4]**

- Module `runs` added.

**[2009/12/06 v2.5]**

- Module `lastpage`: Uses package `atveryend`.
- Module `titleref`: Further commands are disabled during string expansion, imported from package `nameref`.

**[2009/12/07 v2.6]**

- Version date added for package `atveryend`.

**[2009/12/08 v2.7]**

- Module `titleref`: Use of package `getttitlestring`.

**[2010/03/26 v2.8]**

- `\zifrefundefined` added.
- Module `lastpage`: Macros `\zref@iflastpage` and `\ziflastpage` added.
- Module `thepage` added.
- Module `nextpage` added.

**[2010/03/29 v2.9]**

- Module `marks` added (without documentation).
- `\zref@addprop` now adds expanded property to list.
- Useless `\ZREF@ErrorNoLine` removed.

**[2010/04/08 v2.10]**

- Module `xr` remembers the external document name in property `'externaldocument'`.

**[2010/04/15 v2.11]**

- Module `titleref`: Better support of class `memoir`.
- Module `titleref`: Support of theorems.

**[2010/04/17 v2.12]**

- Module `base`: `\zref@newprop` ensures global empty default.
- Module `xr`: Setup options `tozreflabel` and `toptxlabel` added.

**[2010/04/19 v2.13]**

- `\zref@setcurrent` throws an error if the property does not exist (Florent Chervet).
- `\zref@getcurrent` the documentation is fixed (Florent Chervet). Also it returns the empty string in case of errors.
- `\zref@addprop` and `\zref@localaddprop` now take a list of property names (feature request of Florent Chervet).
- Example for `\zref@wrapper@unexpanded` corrected (Florent Chervet).

**[2010/04/22 v2.14]**

- Bug fix for `\zref@getcurrent` second argument wasn't eaten in case of unknown property.
- `\zref@getcurrent` supports `\zref@wrapper@unexpanded`.
- `\zref@wrapper@unexpanded` added for `\ZREF@xr@tlabel`.
- `\zref@extract`, `\zref@extractdefault`, `\zref@getcurrent` are expandable in exact two steps except inside `\zref@wrapper@unexpanded`.

**[2010/04/23 v2.15]**

- `\zexternaldocument` fixed for property 'url' when importing `\new@label` (bug found by Victor Ivrii).
- Two expansion steps also in `\zref@wrapper@unexpanded`.
- Nested calls of `\zref@wrapper@unexpanded` possible.

**[2010/04/28 v2.16]**

- More consequent use of package 'ltxcmds' and 'hologo'.
- Module `pagelayout` added.
- Module `pageattr` added.
- Robustness introduced for non-expandable interface macros.
- Internal change of the data format of property lists (suggestion of Florent Chervet).
- Module `titleref`: Support of environment `description`.

**[2010/05/01 v2.17]**

- `\zref@newprop` throws an error if the property already exists.
- Module `xr`: Bug fix for the case of several `.aux` files (bug found by Victor Ivrii).
- Module `xr`: Property 'urluse' and option `urluse` added.

**[2010/05/13 v2.18]**

- Module `env` added.
- Module `savepos`: `\zref@savepos` added.

**[2010/10/22 v2.19]**

- `\zref@addprop` and `\zref@localaddprop` are limited to one property only (incompatibility to versions v2.13 to v2.18).
- `\zref@addprops` and `\zref@localaddprops` added.
- `\zref@delprop` and `\zref@localdelprop` added.
- `\zref@labelbykv` and `\zkvlabel` (module `user`) with keys `prop`, `list`, `delprop`, `immediate`, `values` added.

[2011/02/12 v2.20]

- Fix for warning in zref-xr.

[2011/03/18 v2.21]

- Fix in module pagelayout for \zlistpagelayout.
- Fix for \zref@localaddprop (probably since v2.19).

## 11 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	
\@@end	2523
\@PackageError	508, 524
\@PackageInfo	1971
\@PackageInfoNoLine	1198, 1274, 1360, 1373, 1385, 1441, 1486
\@PackageWarning	661
\@addtoreset	859, 952
\@auxout	685
\@begintheorem	1910, 1915
\@bsphack	570, 580, 600, 2381
\@caption	1766
\@chapter	1778, 1815
\@currenthref	888
\@currentlabel	883
\@currenvr	2478
\@currenvline	2481, 2484
\@ehc	296, 306, 491, 514, 526, 1516
\@empty	2554, 2557, 2563, 2564, 2565, 2566, 2568
\@esphack	577, 597, 612, 2386
\@for	2652
\@ifclassloaded	1813, 1850
\@ifdefinable	242, 289
\@ifnextchar	530, 1632
\@ifpackageloaded	1870, 1878, 1886, 1905
\@ifstar	495, 1991
\@ifundefined	192, 857, 1643, 2127, 2169
\@input	2188
\@inputcheck	2019, 2020, 2035, 2072, 2074
\@latex@warning	747
\@mainaux	1601
\@namedef	535
\@newl@bel	285
\@nil	1531, 1533, 1731, 1738, 1909, 1910, 2063, 2066, 2246, 2255, 2484, 2487, 2625
\@onelevel@sanitize	422, 440, 505, 533, 1908, 1911, 2538
\@opargbegintheorem	1900
\@part	1772
\@schapter	1796
\@sect	1784
\@spart	1790
\@ssect	1802
\@testdef	1256, 1257, 1420, 1421
\@testopt	1993, 1996, 2006
\@tfor	322, 690
\@undefined	1687, 2245
\@	25, 26, 27, 28, 153, 155, 157, 158, 170, 173, 2154, 2248, 2273, 2283, 2287, 2303
\_	44, 45
<b>A</b>	
\AddLineBeginAux	280
\advance	997, 1313, 1468, 1656, 1967, 2442
\afterassignment	233, 1077, 1081
\AfterLastShipout	994, 1330, 1493
\Alph	7
\anchor	2285
\AtBeginDocument	973, 1166, 1595, 1764, 2591
\AtBeginShipout	956, 1038
\AtEndOfPackage	195
\AtVeryEndDocument	1265, 1429
<b>B</b>	
\beamer@section	1852
\beamer@subsection	1858
\beamer@subsubsection	1864
\begin	23, 57, 100, 106, 156, 172, 2503, 2550, 2555, 2560, 2587, 2592, 2597, 2598, 2610, 2651
\bfseries	876
<b>C</b>	
\c@abspage	959, 1664, 1671
\c@page	997, 1618, 1661
\c@zpage	1661, 1667
\c@zref@unique	863, 1656, 2442, 2444
\ch@pt@c	1819, 1825, 1831
\chapter	24, 30, 32, 61, 82
\ChapterPages	91, 112
\ChapterStart	78, 135, 150, 166
\ChapterStop	85, 148, 165, 184
\chardef	1099, 1114, 1123, 1127

<code>\cleaders</code>	2466	631, 706, 708, 714, 757, 758,	
<code>\cleardoublepage</code>	79, 86	760, 780, 803, 804, 805, 831,	
<code>\clearpage</code>	62	844, 1261, 1425, 1566, 1574,	
<code>\closein</code>	2035	1625, 1626, 1644, 1646, 1649,	
<code>\columnsep</code>	1247	1650, 1651, 1663, 1669, 1674,	
<code>\columnwidth</code>	1246	1675, 1677, 1679, 1680, 1687,	
<code>\comma@entry</code>	343, 344, 346, 352, 378, 379, 381, 387, 584, 586, 590, 1537, 1538, 1539, 1545, 1548	1740, 1743, 1970, 1977, 1989, 2013, 2117, 2119, 2134, 2135, 2152, 2159, 2162, 2176, 2177, 2197, 2294, 2317, 2320, 2501, 2630, 2632, 2634, 2637, 2640, 2645	
<code>\comma@parse</code>	342, 377, 583, 1536		
<code>\count@</code>	1271, 1282, 1283, 1285, 1312, 1313, 1322, 1324, 1325, 1438, 1459, 1461, 1462, 1467, 1468, 1961, 1966, 1977, 1978	<code>\endinput</code>	
<code>\csname</code>	252, 253, 290, 315, 316, 317, 326, 351, 352, 369, 370, 386, 387, 404, 405, 425, 427, 444, 462, 465, 478, 536, 538, 539, 544, 548, 557, 574, 590, 621, 631, 700, 708, 714, 757, 758, 760, 780, 803, 804, 805, 844, 1261, 1425, 1566, 1574, 1626, 1644, 1646, 1649, 1651, 1663, 1669, 1674, 1675, 1677, 1679, 1680, 1687, 1740, 1743, 1970, 1977, 1989, 2013, 2117, 2119, 2134, 2135, 2152, 2159, 2162, 2176, 2177, 2197, 2294, 2317, 2319, 2501, 2630, 2632, 2634, 2637, 2640, 2645	192, 264, 277, 910, 948, 969, 991, 1032, 1068, 1159, 1177, 1341, 1397, 1504, 1615, 1698, 1931, 2334, 2346, 2353, 2366, 2404, 2409, 2421, 2476	
<code>\current@chapid</code>	80, 88	<code>\escapechar</code>	
<b>D</b>		329, 420, 460, 461, 467, 692, 1186, 1213, 1348	
<code>\DeclareBoolOption</code>	1946, 1947, 1948, 1952	<code>\etex@unexpanded</code>	
<code>\DeclareOption</code>	194	561, 789, 809, 2095	
<code>\default</code>	2266, 2306, 2504, 2513, 2517	<code>\evensidemargin</code>	
<code>\DefExpand</code>	2534, 2545	1238	
<code>\define@key</code>	1739, 1742, 1745, 1748, 1949, 2426, 2429, 2432	<code>\Expect</code>	
<code>\descriptionlabel</code>	1808	2500, 2505, 2507, 2512, 2521, 2541, 2543, 2546, 2588, 2593, 2599, 2625	
<code>\detokenize</code>	1726	<code>\ExpectList</code>	
<code>\dftest</code>	167, 174, 175, 176, 177, 178, 179, 180, 181, 182	2499, 2504, 2509, 2513, 2515, 2517, 2518, 2520	
<code>\dimen@</code>	2447, 2453, 2466	<code>\ExpectRuns</code>	
<code>\dimexpr</code>	153, 155, 1325, 2453	2593, 2599	
<code>\directlua</code>	1190, 1352	<code>\externaldocument</code>	
<code>\do</code>	327, 690, 2652	2121, 2163	
<code>\documentclass</code>	4, 39, 68, 272, 2492, 2527, 2581, 2605	<b>F</b>	
<code>\dotfill</code>	169, 173	<code>\fancyhead</code>	
<b>E</b>		51, 54	
<code>\emph</code>	150	<code>\fancyhf</code>	
<code>\end</code>	34, 64, 130, 159, 183, 185, 2522, 2559, 2576, 2577, 2589, 2594, 2600, 2601, 2655, 2658	50, 53	
<code>\endcsname</code>	252, 253, 290, 315, 316, 317, 326, 351, 352, 369, 370, 386, 387, 404, 405, 425, 427, 444, 463, 465, 478, 536, 538, 539, 544, 548, 557, 574, 590, 621,	<code>\fancypagestyle</code>	
		52	
		<code>\filename@area</code>	
		2096	
		<code>\filename@parse</code>	
		2005	
		<code>\foo</code>	
		18, 29, 31, 33	
		<code>\foobar</code>	
		2505, 2507, 2512, 2513, 2520, 2521	
		<code>\footskip</code>	
		1243	
		<code>\frontmatter</code>	
		58, 103	
<b>G</b>		<b>G</b>	
		<code>\g@addto@macro</code>	
		350, 368, 1649, 2615	
		<code>\G@refundefinedtrue</code>	
		746	
		<code>\gdef</code>	
		412, 539, 544, 893, 1261, 1602, 1644, 1646	
		<code>\GetTitleStringDisableCommands</code>	
		1735	
		<code>\GetTitleStringExpand</code>	
		1721	
		<code>\GetTitleStringNonExpand</code>	
		1723	
		<code>\GetTitleStringResult</code>	
		1726	
<b>H</b>		<b>H</b>	
		<code>\hb@xt@</code>	
		2466	
		<code>\headheight</code>	
		1241	
		<code>\headnamereffalse</code>	
		2639	
		<code>\headnamereftrue</code>	
		2644	
		<code>\headsep</code>	
		1242	
		<code>\hfill</code>	
		2454, 2466	
		<code>\hoffset</code>	
		1234	
		<code>\hss</code>	
		2466	
<b>I</b>		<b>I</b>	
		<code>\if@filesw</code>	
		680, 995, 1600, 2375, 2382	
		<code>\if@safe@actives</code>	
		837	
		<code>\ifcase</code>	
		115, 1135, 1537	



```
\ifcename ..... 831, 1625, 1650
\ifeof ..... 2020, 2074
\ifetex@unexpanded ..... 267
\ifheadnameref ..... 1827, 1840
\ifin@ ..... 317
\ifluatex ..... 1182, 1344
\ifnum ..... 476,
1009, 1110, 1120, 1126, 1183,
1312, 1345, 1467, 1509, 1562,
1966, 2043, 2047, 2052, 2360, 2451
\ifodd ..... 124
\ifpdf ..... 2356
\ifx ..... 437, 441, 474,
507, 643, 646, 660, 699, 765,
908, 913, 920, 946, 967, 989,
1030, 1066, 1157, 1175, 1260,
1339, 1396, 1405, 1424, 1502,
1519, 1528, 1532, 1537, 1538,
1539, 1613, 1674, 1696, 1819,
1823, 1889, 1913, 1929, 1977,
2060, 2083, 2088, 2093, 2107,
2154, 2245, 2248, 2273, 2276,
2283, 2287, 2303, 2332, 2344,
2402, 2407, 2419, 2474, 2481, 2637
\ifZREF@found ..... 247, 2264, 2271
\ifZREF@immediate .....
..... 604, 670, 682, 686, 701
\ifZREF@pa@list ..... 1412, 1417
\ifZREF@pl@list ..... 1248, 1253
\ifzref@titleref@expand . . . 1704, 1720
\ifzref@titleref@stripperperiod . . .
..... 1717, 1728
\ifZREF@xr@tooltxlabel . . . . . 2140, 2182
\ifZREF@xr@tozreflabel . . . . . 2126, 2168
\ifZREF@xr@urluse . . . . . 2010, 2289, 2319
\ifZREF@xr@verbose . . . . . 2128, 2170, 2191
\ifZREF@xr@zreflabel . . . . .
..... 1941, 2027, 2041, 2082
\immediate ..... 675, 1601
\in@ ..... 314
\IncludeTests . . . . . 2495, 2530, 2584, 2608
\item . . . . . 107, 111, 113, 121, 125, 127

                                     K
\kern ..... 2462
\kv@define@key 614, 625, 636, 641, 658
\kv@key . . . . . 662, 1531, 1533, 1534, 1548
\kv@parse ..... 659, 1527
\kv@value . . . . . 660, 1528, 1529, 1536
\kvsetkeys . . . . . 603, 1752, 1954, 2438

                                     L
\Label 2613, 2631, 2633, 2636, 2643, 2648
\label ..... 913, 920, 1760, 2552
\leavevmode ..... 2441
\List ..... 2612, 2615, 2657
\LogTests . . . . . 2496, 2531, 2585, 2609
\lst@caption ..... 1892
\lst@label ..... 1889
\lst@MakeCaption ..... 1888
\LT@caption ..... 1880
\ltx@backslashchar .....
..... 713, 1437, 1487, 1972
\ltx@car ..... 1531, 2063
\ltx@carthree ..... 2625
\ltx@cdr ..... 1533, 2066
\ltx@empty 290, 499, 602, 689, 766,
1519, 1705, 1731, 1738, 1819,
1823, 1889, 2002, 2037, 2060,
2066, 2107, 2261, 2481, 2484, 2487
\ltx@firstofone .....
. . . . . 254, 815, 826, 832, 1406, 1407
\ltx@firstoftwo 769, 796, 797, 840, 1011
\ltx@gobble .....
. . . . . 250, 355, 390, 593, 633, 913,
914, 920, 1315, 1470, 1550, 1760
\ltx@gobblethree ..... 921
\ltx@gobbletwo . . . . . 664, 859, 952, 1552
\ltx@ifpackageloaded ..... 2214
\ltx@ifUndefined ..... 229, 249,
257, 410, 825, 867, 1043, 1188,
1350, 1372, 1384, 1403, 1404,
1432, 1475, 1814, 1816, 2348, 2358
\ltx@ifundefined ..... 300, 485,
553, 730, 775, 888, 1195, 1214,
1255, 1357, 1419, 1939, 2199, 2200
\ltx@LocalAppendToMacro .....
. . . . . 385, 403, 619, 629, 1485,
2118, 2158, 2161, 2266, 2269,
2278, 2285, 2291, 2298, 2305, 2317
\ltx@newif ..... 1248, 1412
\ltx@one .....
1313, 1468, 1656, 1961, 1967, 2442
\ltx@ReturnAfterFi ..... 2250
\ltx@secondoftwo .....
311, 754, 767, 797, 834, 838, 1013
\ltx@space ..... 554,
556, 776, 785, 799, 802, 1120,
1126, 1284, 1323, 1511, 1539, 1582
\ltx@undefined ..... 1189, 1351
\ltx@zero . . . . . 476, 1509, 1562, 1632, 1957
\luatexversion ..... 1183, 1345

                                     M
\m@ne ..... 997
\M@sect ..... 1839
\M@TitleReference ..... 2208
\mag ..... 1225
\mainmatter ..... 60, 134
\makeatletter ..... 11, 74,
101, 1990, 2498, 2533, 2553, 2611
\makeatother ..... 16, 99, 2548
\makebox ..... 169, 170
\MakeRobustcommand ..... 232
\marginparsep ..... 1245
\marginparwidth ..... 1244
\mbox ..... 2619
\meaning . . . . . 1910, 2505, 2507, 2512, 2521
\MessageBreak . . . . . 270, 513, 650, 1270,
1282, 1286, 1326, 1437, 1460,
1463, 1487, 1489, 1512, 1515,
1542, 1544, 1583, 1584, 1972,
1973, 2022, 2039, 2044, 2048,
2053, 2109, 2228, 2237, 2350, 2363
```

<b>N</b>	
<code>\NeedsTeXFormat</code> . . . . .	3, 188, 220, 904, 942, 963, 983, 1026, 1062, 1153, 1171, 1335, 1498, 1591, 1609, 1692, 1925, 2328, 2340, 2398, 2415, 2470, 2490, 2526, 2580, 2604
<code>\newcommand</code> . . . . .	18, 78, 85, 91, 167, 912, 919, 925, 1076, 1093, 1094, 1163, 1506, 1953, 2389, 2392, 2438, 2534, 2540
<code>\newcount</code> . . . . .	1956
<code>\newcounter</code> . . . . .	6, 860, 953, 1621
<code>\newif</code> . . . . .	247, 670, 1704, 1717, 1941
<code>\newlabel</code> . . . . .	2145, 2156, 2187
<code>\newmarks</code> . . . . .	1515
<code>\newpage</code> . . . . .	143, 2656
<code>\nfss@text</code> . . . . .	876
<code>\nofiles</code> . . . . .	2491
<code>\NR@temp</code> . . . . .	1822, 1823
<code>\number</code> . . . . .	94, 109, 863, 868, 1044, 1218, 1219, 1596, 1618, 1638, 1664, 1671, 2444, 2593, 2599
<code>\numexpr</code> . . . . .	94, 109, 115, 870, 1046, 1107, 1596, 1638, 2102, 2113, 2147, 2232, 2241, 2451
<b>O</b>	
<code>\oddsidemargin</code> . . . . .	1237
<code>\on@line</code> . . . . .	1598, 2037
<code>\openin</code> . . . . .	2019
<b>P</b>	
<code>\PackageError</code> . . . . .	258, 269, 294, 304, 489, 1510, 2349, 2361
<code>\PackageInfo</code> . . . . .	291, 520, 1599, 2026, 2038, 2129, 2171, 2192
<code>\PackageWarning</code> . . . . .	345, 363, 380, 398, 585, 649, 1540, 1581, 2021
<code>\PackageWarningNoLine</code>	2108, 2227, 2236
<code>\page</code> . . . . .	2269, 2504, 2513, 2515, 2517
<code>\pagestyle</code> . . . . .	49
<code>\paperheight</code> . . . . .	1227
<code>\paperwidth</code> . . . . .	1226
<code>\par</code> . . . . .	2616
<code>\part</code> . . . . .	2637
<code>\pdf@escapehex</code> . . . . .	1409
<code>\pdf@strcmp</code> . . . . .	476
<code>\pdf@unescapehex</code> . . . . .	1410
<code>\pdfhorigin</code> . . . . .	1206, 1232
<code>\pdflastxpos</code> . . . . .	2371
<code>\pdflastypos</code> . . . . .	2372
<code>\pdfpageattr</code> . . . . .	1366, 1374, 1380
<code>\pdfpageheight</code> . . . . .	1205, 1231
<code>\pdfpagesattr</code> . . . . .	1367, 1386, 1392
<code>\pdfpagewidth</code> . . . . .	1204, 1230
<code>\pdfsavepos</code>	2350, 2362, 2376, 2445, 2459
<code>\pdftexversion</code> . . . . .	2360
<code>\pdfvorigin</code> . . . . .	1207, 1233
<code>\ProcessOptions</code> . . . . .	217
<code>\protect</code> . . . . .	746
<code>\protected</code> . . . . .	238
<code>\protected@write</code> . . . . .	685
<code>\providecommand</code> . . . . .	281, 1594, 1938, 2435, 2436, 2437
<code>\ProvidesPackage</code> . . . . .	189, 221, 905, 943, 964, 984, 1027, 1063, 1154, 1172, 1336, 1499, 1592, 1610, 1693, 1926, 2329, 2341, 2399, 2416, 2471
<b>R</b>	
<code>\read</code> . . . . .	2072
<code>\refstepcounter</code> . . . . .	975
<code>\renewcommand</code> . . . . .	7, 46, 862
<code>\RequirePackage</code> . . . . .	191, 196, 223, 224, 225, 226, 227, 230, 266, 271, 279, 907, 945, 950, 966, 986, 987, 988, 1029, 1034, 1035, 1065, 1070, 1071, 1072, 1073, 1156, 1161, 1162, 1174, 1179, 1180, 1181, 1338, 1343, 1379, 1391, 1399, 1400, 1401, 1501, 1612, 1617, 1695, 1700, 1701, 1928, 1933, 1934, 2331, 2343, 2355, 2401, 2406, 2411, 2418, 2423, 2425, 2473
<code>\reset@font</code> . . . . .	876
<code>\rightarrow</code> . . . . .	45
<code>\romannumeral</code> . . . . .	552, 774, 795, 1520, 1968, 1977, 1978
<b>S</b>	
<code>\section</code> . . . . .	63, 137, 145, 2551
<code>\setcounter</code> . . . . .	955
<code>\setlength</code> . . . . .	2447
<code>\SetupKeyvalOptions</code> . . . . .	1942
<code>\space</code> . . . . .	748, 1374, 1386, 2040, 2041, 2045, 2049, 2054, 2350, 2362
<code>\stepcounter</code> . . . . .	19, 957, 1623, 1624
<code>\stockheight</code> . . . . .	1229
<code>\stockwidth</code> . . . . .	1228
<b>T</b>	
<code>\tableofcontents</code> . . . . .	59, 132
<code>\Test</code> . . . . .	2540, 2556, 2558, 2563, 2565, 2567, 2571, 2572, 2629, 2653
<code>\textbf</code> . . . . .	2551, 2561, 2574
<code>\textheight</code> . . . . .	1240
<code>\textit</code> . . . . .	2551, 2562, 2574
<code>\textwidth</code> . . . . .	1239
<code>\the</code> . . . . .	13, 153, 155, 429, 444, 460, 590, 596, 712, 726, 870, 959, 1003, 1040, 1046, 1107, 1254, 1274, 1281, 1282, 1283, 1285, 1322, 1324, 1325, 1380, 1392, 1418, 1442, 1458, 1459, 1461, 1714, 2096, 2102, 2113, 2147, 2232, 2241, 2371, 2372
<code>\thechapter</code> . . . . .	14
<code>\thefoo</code> . . . . .	7, 12, 20
<code>\thetype</code> . . . . .	2159
<code>\thepage</code> . . . . .	43, 44, 45, 683, 687, 748, 884, 1662
<code>\thezpage</code> . . . . .	16, 1662, 1668

<code>\thezref@unique</code> .....	<code>\znextpage</code> .....
10, <a href="#">862</a> , 1659, 1660, 1667, 1668,	13, 51, 54, 1090
1670, 2446, 2448, 2451, 2452, 2460	<code>\znextpagesetup</code> .....
<code>\title</code> .....	14, 42, 1076
2278, 2307	<code>\znonextpagename</code> .....
<code>\toks@</code> .....	46, 1093, 1141
423, 429, 443,	<code>\zpageref</code> .....
444, 582, 589, 590, 596, 710,	11, 126, <a href="#">934</a>
712, 725, 726, 998, 1003, 1270,	<code>\zposx</code> .....
1274, 1280, 1281, 1437, 1442,	17, 153, <a href="#">2389</a> , 2451, 2452
1457, 1458, 1708, 1714, 2081, 2096	<code>\zposy</code> .....
<code>\topmargin</code> .....	17, 155, <a href="#">2389</a>
1236	<code>\zref</code> .....
<code>\TR@TitleReference</code> ..	11, 25, 26, 27, 28, 112,
2203, 2263, 2302	114, 123, 128, 129, 139, <a href="#">925</a> , 935
<code>\ttl@sect@i</code> .....	<code>\ZREF@@@delprop</code> ...
1872	434, <a href="#">436</a> , 471, <a href="#">473</a>
<b>U</b>	
<code>\unexpanded</code> .....	<code>\ZREF@@@newprop</code> .....
270, 275	538, 542
<code>\UniqueCounterCall</code> .....	<code>\ZREF@delprop</code> .....
1091	424, <a href="#">433</a> , 447, 464, <a href="#">470</a> , 480
<code>\UniqueCounterNew</code> .....	<code>\ZREF@extract</code> .....
1074	778, <a href="#">784</a>
<code>\url</code> .....	<code>\ZREF@makeperpage</code> ..
2298	1632, 1638, 1642
<code>\urluse</code> .....	<code>\ZREF@newprop</code> ....
2293	516, 527, 530, <a href="#">534</a>
<code>\usepackage</code> .....	<code>\ZREF@perpage@step</code> .....
9, 41, 48, 70, 72, 2493, 2494,	1647, <a href="#">1655</a>
2528, 2529, 2582, 2583, 2606, 2607	<code>\zref@addprop</code> .....
<b>V</b>	
<code>\value</code> .....	5, 76,
13, 1040, 1254, 1418	359, 960, 972, 1037, 1220, 1381,
<code>\verb</code> .....	1393, 1548, 1703, 2337, 2412, 2519
173	<code>\zref@addprops</code> 15, <a href="#">340</a> , 885, 1620, 2373
<code>\voffset</code> .....	<code>\ZREF@addtoks</code> .....
1235	724
<b>W</b>	
<code>\write</code> .....	<code>\ZREF@base@ok</code> .....
674, 675, 1601	901, 908,
<b>X</b>	
<code>\x</code> .....	946, 967, 989, 1030, 1066, 1157,
330,	1175, 1339, 1502, 1613, 1696,
335, 693, 698, 845, 847, 1133,	1929, 2332, 2344, 2402, 2419, 2474
1149, 1215, 1222, 1273, 1276,	<code>\ZREF@call</code> 1099, 1114, 1123, 1127, 1135
1279, 1311, 1440, 1445, 1456,	<code>\zref@default</code> ...
1466, 1564, 1569, 1572, 1578,	8, 530, 776, 873, <a href="#">875</a>
1637, 1640, 1907, 1908, 1913,	<code>\ZREF@delprop</code> .....
2080, 2083, 2088, 2093, 2103,	412, 415, <a href="#">417</a> , 452, 455, <a href="#">457</a>
2106, 2116, 2117, 2148, 2153,	<code>\zref@delprop</code> .....
2545, 2546, 2624, 2625, 2652, 2653	411, <a href="#">451</a>
<code>\XR@ext</code> .....	<code>\ZREF@df@dot</code> .....
1939	2433, 2437, 2466
<b>Y</b>	
<code>\y</code> .....	<code>\ZREF@df@min</code> .....
1909, 1910	2430, 2435, 2453
<b>Z</b>	
<code>\z</code> .....	<code>\ZREF@df@unit</code> .....
1910, 1911, 1913	2427, 2436, 2447
<code>\z@</code> .....	<code>\ZREF@dotfill</code> .....
2462	2449, 2456, <a href="#">2465</a>
<code>\Z@D@page</code> .....	<code>\ZREF@ENV@line</code> .....
1094	2484, <a href="#">2487</a>
<code>\Z@L@LastPage</code> .....	<code>\zref@env@line</code> .....
1000	2479, <a href="#">2480</a>
<code>\Z@L@main</code> .....	<code>\ZREF@extract</code> ....
999	773, 790, 793, 823
<code>\Z@L@ZREF@temp</code> 602, 606, 609, 620, 630	<code>\zref@extract</code> ....
<code>\zdotfill</code> .....	7, 95, 96, 109,
18, 170, 173, <a href="#">2439</a>	140, <a href="#">773</a> , <a href="#">793</a> , 818, 823, 932,
<code>\zdotfillsetup</code> .....	1049, 1145, 1285, 1324, 1325,
18, <a href="#">2438</a>	1449, 1479, 1667, 1668, 1761,
<code>\zexternaldocument</code> .....	2390, 2393, 2567, 2571, 2572, 2624
18, <a href="#">1984</a>	<code>\ZREF@extractdefault</code> <a href="#">794</a> , 810, 813, 822
<code>\ziflastpage</code> .....	<code>\zref@extractdefault</code> .
12, <a href="#">1016</a>	7, 116, 117,
<code>\zifrefundefined</code> .....	786, <a href="#">813</a> , 817, 822, 1009, 1010,
7, <a href="#">732</a>	1103, 1118, 1164, 1670, 2201,
<code>\zkvlabel</code> .....	2204, 2205, 2209, 2210, 2213,
919	2215, 2216, 2218, 2220, 2563, 2565
<code>\zlabel</code> .....	<code>\ZREF@false</code> .....
11, 83, 104, 138, 146, <a href="#">912</a>	646, <a href="#">656</a>
<code>\zlistpageattr</code> .....	<code>\ZREF@foundfalse</code> .....
1413	2262
<code>\zlistpagelayout</code> .....	<code>\ZREF@foundtrue</code> .....
15, <a href="#">1249</a>	2309
<code>\zmakeperpage</code> .....	<code>\ZREF@getcurrent</code> ..
16, <a href="#">1630</a>	551, 562, 565, 821
	<code>\zref@getcurrent</code> .....
	6, <a href="#">565</a> , 816, 821, 2556, 2558
	<code>\zref@hex</code> ....
	1380, 1392, 1406, 1409
	<code>\ZREF@IfDefinable</code> .....
	241, 732, 934, 937, 1016,
	1054, 1090, 1249, 1413, 1630,
	1684, 1751, 1754, 1984, 2379, 2439
	<code>\ZREF@iflastpage</code> ...
	1017, <a href="#">1019</a> , 1019
	<code>\zref@iflastpage</code> .....
	12, <a href="#">1008</a> , 1022

<code>\zref@iflistcontainsprop</code> . . . . .	6, 309, 344, 362, 379, 397, 617, 627	<code>\ZREF@nil</code> 544, 766, 805, 2073, 2079,	2085, 2090, 2100, 2116, 2145,
<code>\zref@iflistundefined</code> . . . . .	6, 288, 299, 303, 310	2153, 2244, 2251, 2260, 2263, 2302	
<code>\zref@ifpropundefined</code> . . . . .	6, 484, 488, 518, 584, 797, 1320, 1561, 2256	<code>\ZREF@NOVALUE</code> . . . . .	772
<code>\ZREF@ifrefcontainsprop</code> . . . . .	756, 764	<code>\ZREF@novalue</code> . . . . .	765, 766, 772
<code>\zref@ifrefcontainsprop</code> . . . . .	8, 752, 1322, 2217, 2313, 2314	<code>\ZREF@np@call@next</code> . . . . .	1085, 1089, 1144
<code>\ZREF@ifrefundefined</code> . . . . .	734, 737, 1100, 1111, 1121	<code>\ZREF@np@call@nonext</code> . . . . .	1082, 1088, 1140
<code>\zref@ifrefundefined</code> . . . . .	7, 729, 739, 745, 753, 796, 1112, 1283, 1461, 1660, 2448, 2569, 2621	<code>\ZREF@np@call@unknown</code> . . . . .	1078, 1087, 1136
<code>\ZREF@immediatefalse</code> . . . . .	647	<code>\ZREF@np@setup@i</code> . . . . .	1077, 1080
<code>\ZREF@immediatetrue</code> . . . . .	644, 673	<code>\ZREF@np@setup@ii</code> . . . . .	1081, 1084
<code>\ZREF@label</code> 572, 596, 606, 609, 679, 1003		<code>\ZREF@number</code> . . . . .	867, 1449, 1453, 1508
<code>\zref@label</code> . . . . .	6, 566, 916, 2614	<code>\ZREF@org@@begintheorem</code> . . . . .	1917
<code>\zref@labelbykv</code> . . . . .	599, 923	<code>\ZREF@org@@caption</code> . . . . .	1768
<code>\zref@labelbylist</code> . . . . .	7, 567, 569, 1040, 1659, 2384	<code>\ZREF@org@@chapter</code> . . . . .	1780, 1836
<code>\zref@labelbyprops</code> . . . . .	7, 88, 579, 1098, 2446, 2460	<code>\ZREF@org@@opargbegintheorem</code> . . . . .	1902
<code>\zref@listexists</code> . . . . .	6, 302, 321, 341, 360, 376, 395, 418, 458, 571	<code>\ZREF@org@@part</code> . . . . .	1774
<code>\zref@listforloop</code> . . . . .	320, 626	<code>\ZREF@org@@schapter</code> . . . . .	1798
<code>\zref@listpageattr</code> . . . . .	1413	<code>\ZREF@org@@sect</code> . . . . .	1786
<code>\zref@listpagelayout</code> . . . . .	1249	<code>\ZREF@org@@spart</code> . . . . .	1792
<code>\zref@localaddprop</code> . . . . .	5, 394, 2511, 2514	<code>\ZREF@org@@ssect</code> . . . . .	1804
<code>\zref@localaddprops</code> . . . . .	375	<code>\ZREF@org@beamer@section</code> . . . . .	1854
<code>\zref@localdelprop</code> . . . . .	414, 454, 638	<code>\ZREF@org@beamer@subsection</code> . . . . .	1860
<code>\ZREF@mainlist</code> . . . . .	567, 879, 882, 885, 960, 972, 1703, 2337	<code>\ZREF@org@beamer@subsubsection</code> . . . . .	1866
<code>\ZREF@makeperpage@opt</code> . . . . .	1632, 1635	<code>\ZREF@org@descriptionlabel</code> . . . . .	1810
<code>\ZREF@MARKS@DefineProp</code> . . . . .	1524, 1525, 1526, 1560	<code>\ZREF@org@lst@MakeCaption</code> . . . . .	1895
<code>\zref@marks@register</code> . . . . .	1506, 1511, 1543, 1582	<code>\ZREF@org@LT@ccoption</code> . . . . .	1881
<code>\ZREF@name</code> . . . . .	228, 258, 269, 291, 294, 304, 345, 363, 380, 398, 489, 508, 520, 524, 585, 649, 661, 1510, 2349, 2361	<code>\ZREF@org@M@sect</code> . . . . .	1845
<code>\ZREF@NAME@bot</code> . . . . .	1539, 1559	<code>\ZREF@org@refstepcounter</code> . . . . .	977
<code>\ZREF@NAME@first</code> . . . . .	1538, 1558	<code>\ZREF@org@stepcounter</code> . . . . .	1623, 1628
<code>\ZREF@NAME@top</code> . . . . .	1537, 1557	<code>\ZREF@org@testdef</code> . . . . .	1256, 1258, 1420, 1422
<code>\zref@newlabel</code> . . . . .	7, 281, 284, 719, 2100, 2186	<code>\ZREF@org@thepage</code> . . . . .	683, 687
<code>\zref@newlist</code> . . . . .	5, 287, 882, 993, 1036, 1534, 1619, 2370, 2508	<code>\ZREF@org@ttl@sect@i</code> . . . . .	1874
<code>\ZREF@newprop</code> . . . . .	497, 500, 503	<code>\ZREF@org@write</code> . . . . .	674, 675
<code>\zref@newprop</code> . . . . .	6, 12, 13, 14, 75, 494, 883, 884, 887, 894, 898, 959, 971, 1217, 1380, 1392, 1565, 1573, 1618, 1935, 1936, 1937, 2257, 2371, 2372, 2478, 2479, 2506, 2554	<code>\ZREF@P</code> . . . . .	504, 505, 507, 509, 518, 521, 525, 535, 536, 538, 539, 540, 544, 690, 694, 695, 704, 708, 713, 714
<code>\ZREF@NewPropAnchor</code> . . . . .	886, 1986, 2336	<code>\ZREF@pa@AfterLastShipout</code> . . . . .	1416, 1494
<code>\ZREF@NewPropTheotype</code> . . . . .	897, 2157	<code>\ZREF@pa@AtVeryEnd</code> . . . . .	1429, 1432, 1485
<code>\ZREF@NewPropTitle</code> . . . . .	892, 1702, 1987	<code>\ZREF@pa@ListPage</code> . . . . .	1439, 1455
<code>\ZREF@nextpage</code> . . . . .	1091, 1095	<code>\ZREF@pa@listtrue</code> . . . . .	1414
		<code>\ZREF@page@max</code> . . . . .	1254, 1312, 1418, 1467
		<code>\zref@pageattr</code> . . . . .	1447
		<code>\zref@pageattr@used</code> . . . . .	1452
		<code>\ZREF@pagenum@last</code> . . . . .	1117, 1120
		<code>\ZREF@pagenum@this</code> . . . . .	1102, 1107, 1110, 1120, 1126
		<code>\ZREF@par</code> . . . . .	507, 532
		<code>\ZREF@param</code> . . . . .	421, 422, 441, 459, 476, 615, 616, 617, 621, 642, 643, 646, 651
		<code>\ZREF@patch</code> . . . . .	248, 974, 1765, 1771, 1777, 1783, 1789, 1795, 1801, 1807, 1838, 1851, 1857, 1863, 1871, 1879, 1887, 1899, 1914
		<code>\zref@pdfpageattr</code> . . . . .	1376, 1462
		<code>\zref@pdfpageattr@used</code> . . . . .	1377
		<code>\zref@pdfpagesattr</code> . . . . .	1388, 1477, 1488
		<code>\zref@pdfpagesattr@used</code> . . . . .	1389, 1482
		<code>\ZREF@pl@AfterLastShipout</code> . . . . .	1252, 1331
		<code>\ZREF@pl@AtVeryEnd</code> . . . . .	1265, 1268

\ZREF@pl@ListEntry ... 1287, 1288,  
1289, 1290, 1291, 1292, 1293,  
1294, 1295, 1296, 1297, 1298,  
1299, 1300, 1301, 1302, 1303,  
1304, 1305, 1306, 1307, 1308, 1319  
\ZREF@pl@ListPage ..... 1272, 1278  
\ZREF@pl@listtrue ..... 1250  
\zref@prop 323, 331, 332, 336, 627, 631  
\zref@propexists .... 6, 343, 361,  
378, 396, 487, 547, 616, 637, 926  
\ZREF@refname@next .....  
..... 1105, 1112, 1121, 1145  
\ZREF@refname@this .....  
..... 1097, 1098, 1100, 1103  
\ZREF@RefPrefix . . 283, 285, 1260, 1424  
\ZREF@refused ..... 742, 744  
\zref@refused ..... 7,  
738, 741, 931, 938, 1020, 1021,  
1052, 1167, 1453, 1483, 1759, 2620  
\zref@require@unique .....  
..... 10, 856, 1622, 2424  
\ZREF@Robust ... 231, 237, 243, 284,  
287, 302, 309, 340, 359, 375,  
394, 411, 414, 451, 454, 487,  
494, 546, 566, 569, 579, 599,  
671, 741, 814, 830, 856, 872,  
878, 1051, 1452, 1482, 1706, 1719  
\ZREF@SavedEscapechar ..... 460, 467  
\zref@savepos ..... 17, 2374, 2383  
\ZREF@savepos@ok ..... 2395, 2407  
\zref@setcurrent .....  
..... 6, 81, 540, 546, 666, 976  
\zref@setdefault ..... 8, 872, 875  
\zref@setmainlist ..... 8, 878  
\ZREF@STAR ..... 1532, 1556  
\ZREF@stripperperiod ..... 1730, 1738  
\ZREF@temp ..... 193, 200, 201,  
202, 203, 204, 205, 206, 207,  
208, 209, 210, 211, 212, 213,  
214, 215, 216, 232, 233, 439,  
440, 441, 689, 710, 711, 719,  
1187, 1204, 1205, 1206, 1207,  
1211, 1225, 1226, 1227, 1228,  
1229, 1230, 1231, 1232, 1233,  
1234, 1235, 1236, 1237, 1238,  
1239, 1240, 1241, 1242, 1243,  
1244, 1245, 1246, 1247, 1259,  
1260, 1349, 1366, 1367, 1371,  
1382, 1394, 1396, 1402, 1403,  
1404, 1405, 1423, 1424, 1531,  
1532, 1960, 1970, 1973, 1977, 2444  
\ZREF@TempName .. 1507, 1519, 1520,  
1522, 1548, 1561, 1565, 1573, 1584  
\ZREF@TempNum .....  
1508, 1509, 1513, 1520, 1562, 1575  
\zref@thepage ..... 13, 1048, 1057  
\zref@thepage@name .....  
..... 13, 1043, 1049, 1052, 1106  
\zref@thepage@refused ... 1051, 1056  
\ZREF@titleref ..... 1755, 1757  
\zref@titleref@cleanup .. 1706, 1746  
\zref@titleref@current .....  
..... 892, 1725, 1729, 1730, 1749  
\ZREF@titleref@hook .....  
..... 1705, 1709, 1713, 1736  
\zref@titleref@setcurrent .....  
..... 1719, 1767, 1773,  
1779, 1785, 1791, 1797, 1803,  
1809, 1817, 1820, 1824, 1828,  
1830, 1841, 1843, 1853, 1859,  
1865, 1873, 1882, 1891, 1901, 1916  
\zref@titleref@stripperiodtrue 1718  
\ZREF@true ..... 643, 657  
\ZREF@u@getcurrent ..... 560  
\zref@unhex .... 1407, 1410, 1448, 1478  
\ZREF@UpdatePdfTeX ... 246, 2352, 2365  
\ZREF@wrapper@babel ..... 847, 853  
\zref@wrapper@babel .. 9, 140, 734,  
742, 830, 916, 923, 927, 1017, 1755  
\zref@wrapper@immediate .....  
..... 10, 87, 605, 671, 1002, 1039  
\ZREF@wrapper@unexpanded ... 814, 828  
\zref@wrapper@unexpanded .....  
..... 10, 815, 820, 825, 2196, 2542  
\ZREF@wu@extract ..... 788, 818  
\ZREF@wu@extractdefault ... 808, 817  
\ZREF@wu@getcurrent ..... 560, 816  
\ZREF@X ..... 496, 499, 536  
\zref@xr@ ..... 1950  
\ZREF@xr@AddUrl ..... 1962, 1965  
\ZREF@xr@input ..... 2093, 2188  
\ZREF@xr@AddURL .... 1958, 2011, 2290  
\ZREF@xr@checkfile .. 2015, 2018, 2068  
\ZREF@xr@checkkey ..... 2246, 2255  
\ZREF@xr@checklist ..... 2116, 2244  
\zref@xr@ext ..... 19, 1938, 2006  
\ZREF@xr@externaldocument .....  
..... 1993, 1996, 2000  
\ZREF@xr@externalfile .....  
..... 2003, 2004, 2122, 2164  
\ZREF@xr@file ... 2004, 2019, 2022,  
2028, 2039, 2062, 2110, 2229, 2238  
\ZREF@xr@filelist ..... 2002,  
2060, 2063, 2065, 2066, 2094, 2095  
\ZREF@xr@found . 2030, 2040, 2102, 2147  
\ZREF@xr@graburl ..... 2006, 2008  
\ZREF@xr@ignored@empty .....  
..... 2031, 2043, 2045, 2112, 2113  
\ZREF@xr@ignored@ltx .....  
..... 2033, 2052, 2054, 2240, 2241  
\ZREF@xr@ignored@zref .....  
..... 2032, 2047, 2049, 2231, 2232  
\ZREF@xr@line .. 2072, 2073, 2085, 2090  
\ZREF@xr@list ..... 2106, 2107  
\ZREF@xr@ltx@ignorewarning .... 2235  
\ZREF@xr@newlabel ..... 2088, 2187  
\ZREF@xr@prefix ..... 2001, 2101,  
2137, 2141, 2146, 2172, 2179, 2183  
\ZREF@xr@process@label .. 2090, 2145  
\ZREF@xr@process@zreflabel 2085, 2100  
\ZREF@xr@processfile .... 2018, 2071  
\ZREF@xr@processline .... 2073, 2079

<code>\ZREF@xr@refname</code> .....	<code>\ZREF@xr@zref@ignorewarning</code> .....
2101, 2127, 2134, 2146, 2169, 2176	2137, 2179, <u>2226</u>
<code>\ZREF@xr@relax</code> .....	<code>\ZREF@xr@zref@newlabel</code> ..
2189, 2276	2083, 2186
<code>\ZREF@xr@scanparams</code> .....	<code>\ZREF@xr@zreflabelfalse</code> .....
2151, <u>2260</u>	1992
<code>\ZREF@xr@scantitleref</code> ...	<code>\ZREF@xr@zreflabeltrue</code> .....
2263, <u>2302</u>	1995
<code>\ZREF@xr@temp</code> .....	<code>\ZREF@zref</code> .....
2275, 2276	927, 930
<code>\ZREF@xr@tempname</code> 2104, 2105, 2125,	<code>\zrefused</code> 11, 92, 93, 161, 162, 163, <u>937</u>
2130, 2141, 2149, 2150, 2167, 2183	<code>\zruns</code> .....
<code>\ZREF@xr@temprefname</code> .....	15, <u>1594</u> , 2588, 2593, 2599
..... 2105, 2117, 2119,	<code>\zsavepos</code> .....
2135, 2150, 2152, 2159, 2162, 2177	17, 157, 158, <u>2379</u>
<code>\ZREF@xr@theURL</code> .....	<code>\zthepage</code> .....
1968, 1970, 1972, 1978, 2013, 2294	13, <u>1054</u>
<code>\ZREF@xr@tolabel</code> ...	<code>\ztitleref</code> .....
2141, 2183, <u>2190</u>	17, <u>1754</u> , 2617
<code>\ZREF@xr@URL</code> ...	<code>\ztitlerefsetup</code> .....
1956, 1966, 1967, 1968	17, <u>1739</u>
<code>\ZREF@xr@url</code> ...	<code>\ztotpages</code> .....
2009, 2011, 2012, <u>2320</u>	14, 124, <u>1163</u>
<code>\ZREF@xr@urlcheck</code> ..	<code>\zunknownnextpagename</code> .
2125, 2167, <u>2312</u>	13, 1094, 1137
	<code>\zunmakeperpage</code> .....
	16, <u>1684</u>
	<code>\zxrsetup</code> .....
	19, <u>1953</u>