

The zref package

Heiko Oberdiek
<heiko.oberdiek at gmail.com>

2010/05/01 v2.17

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

1	Introduction	3
1.1	Standard \LaTeX behaviour	4
1.2	Basic idea	4
1.3	Interfaces	4
2	Interface for programmers	5
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
3	User interface	10
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>	17
3.15	Module <code>xr</code>	18

4	ToDo	19
5	Example	19
6	Implementation	21
6.1	Package <code>zref</code>	21
6.1.1	Identification	21
6.1.2	Load basic module	21
6.1.3	Process options	21
6.2	Module <code>base</code>	22
6.2.1	Prefixes	22
6.2.2	Identification	22
6.2.3	Utilities	22
6.2.4	Check for ϵ - <code>T_EX</code>	23
6.2.5	Auxiliary file stuff	24
6.2.6	Property lists	24
6.2.7	Properties	26
6.2.8	Reference generation	28
6.2.9	Reference querying and extracting	30
6.2.10	Compatibility with <code>babel</code>	32
6.2.11	Unique counter support	33
6.2.12	Utilities	33
6.2.13	Setup	33
6.3	Module <code>user</code>	34
6.4	Module <code>abspage</code>	35
6.5	Module <code>counter</code>	35
6.6	Module <code>lastpage</code>	36
6.7	Module <code>thepage</code>	37
6.8	Module <code>nextpage</code>	37
6.9	Module <code>totpages</code>	39
6.10	Module <code>pagelayout</code>	39
6.11	Module <code>pageattr</code>	42
6.12	Module <code>marks</code>	45
6.13	Module <code>runs</code>	47
6.14	Module <code>perpage</code>	47
6.15	Module <code>titleref</code>	49
6.15.1	Implementation	49
6.15.2	User interface	51
6.15.3	Patches for section and caption commands	51
6.15.4	Environment description	52
6.15.5	Class <code>memoir</code>	52
6.15.6	Class <code>beamer</code>	53
6.15.7	Package <code>titlesec</code>	53
6.15.8	Package <code>longtable</code>	53
6.15.9	Package <code>listings</code>	54
6.15.10	Theorems	54
6.16	Module <code>xr</code>	54
6.17	Module <code>hyperref</code>	62
6.18	Module <code>savepos</code>	62
6.18.1	Identification	63
6.18.2	Availability	63
6.18.3	Setup	63
6.18.4	User macros	63
6.19	Module <code>abspos</code>	64
6.19.1	Identification	64
6.20	Module <code>dotfill</code>	64

7	Test	65
7.1	<code>\zref@localaddprop</code>	65
7.2	Module <code>base</code>	66
7.3	Module <code>runs</code>	67
7.4	Module <code>titleref</code>	67
8	Installation	68
8.1	Download	68
8.2	Bundle installation	69
8.3	Package installation	69
8.4	Refresh file name databases	70
8.5	Some details for the interested	70
9	References	70
10	History	71
	[2006/02/20 v1.0]	71
	[2006/05/03 v1.1]	71
	[2006/05/25 v1.2]	71
	[2006/09/08 v1.3]	71
	[2007/01/23 v1.4]	71
	[2007/02/18 v1.5]	71
	[2007/04/06 v1.6]	71
	[2007/04/17 v1.7]	71
	[2007/04/22 v1.8]	71
	[2007/05/02 v1.9]	71
	[2007/05/06 v2.0]	72
	[2007/05/28 v2.1]	72
	[2008/09/21 v2.2]	72
	[2008/10/01 v2.3]	72
	[2009/08/07 v2.4]	72
	[2009/12/06 v2.5]	72
	[2009/12/07 v2.6]	72
	[2009/12/08 v2.7]	72
	[2010/03/26 v2.8]	72
	[2010/03/29 v2.9]	72
	[2010/04/08 v2.10]	73
	[2010/04/15 v2.11]	73
	[2010/04/17 v2.12]	73
	[2010/04/19 v2.13]	73
	[2010/04/22 v2.14]	73
	[2010/04/23 v2.15]	73
	[2010/04/28 v2.16]	73
	[2010/05/01 v2.17]	74
11	Index	74

1 Introduction

Standard L^AT_EX'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^AT_EX 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^AT_EX behaviour

References are created by the `\label` command:

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

Now L^AT_EX 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^AT_EX 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^AT_EX and other packages.

titleref stores its title data into the first entry in the list. L^AT_EX 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^AT_EX3 that will need some time before its first release. Thus I have implemented it as L^AT_EX 2_ε package without disturbing the existing L^AT_EX 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 packages 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

^{exp} means that the implementation of the marked macro is expandable. ^{exp2} 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^AT_EX'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 `\ref@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><empty></code>
	page	main	<code><empty></code>
abspage	abspage	main	0
counter	counter	main	<code><empty></code>
hyperref	anchor	main	<code><empty></code>
	url		<code><empty></code>
pageattr	pdfpageattr	thepage	...
	pdfpagesattr	LastPage	...
pagelayout ¹	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><empty></code>
	abspage	perpage	0
savepos	posx	savepos	0
	posy	savepos	0
titleref	title	main	<code><empty></code>
xr	anchor		<code><empty></code>
	externaldocument		<code><empty></code>
	thetype		<code><empty></code>
	title		<code><empty></code>
	url		<code><empty></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.

¹Module `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 ε -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">• robust, unless the extracted values are fragile,• no babel shorthand support.
<code>exp2</code>	Expandable like <code>exp</code> and: <ul style="list-style-type: none">• expandable in exact two steps.

The basic user interface of the package without modules are commands that mimic the standard L^AT_EX 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}[2010/05/01]
10
11 \makeatletter
12 \zref@newprop{thefoo}{\thefoo}
13 \zref@newprop{valuefoo}{\the\value{foo}}
14 \zref@newprop{chapter}{\thechapter}
15 \zref@addprop{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`

`\znextpage`

Macro `\znextpage` 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 `\znextpage` 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 {<unknown>} {<no next>} {<next>}</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}[2010/05/01]
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^{exp}</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 \TeX engine like `pdf \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\TeX</code> , <code>Lua\TeX</code>
<code>\pdfpageheight</code>	<code>pdfpageheight</code>	<code>pdf\TeX</code> , <code>Lua\TeX</code>
<code>\pdfhorigin</code>	<code>pdfhorigin</code>	<code>pdf\TeX</code> , <code>Lua\TeX</code>
<code>\pdfvorigin</code>	<code>pdfvorigin</code>	<code>pdf\TeX</code> , <code>Lua\TeX</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.

3.8 Module marks

ToDo.

3.9 Module runs

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

`\zrunsexp`

Prints the the total number of \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^AT_EX 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^AT_EX 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 `<refname>`, 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 `<refname>` to the location where the command is executed.

`\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.

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 `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^AT_EX 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_EX memory.

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

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

If the $\langle url \rangle$ 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  $\langle *example \rangle$ 
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{chaptile}{#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 \zpageref{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 pdf \TeX 's “savepos” feature. The position on the page is not known before the page is constructed and shipped out. Therefore the position ist stored in references and are available for calculations in the next \LaTeX 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^AT_EX is notified, if the references are not yet available and L^AT_EX 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*{\dfptest}[1]{%
168   #1&
169   [\makebox[#{1}]{\dotfill}]&
170   [\makebox[#{1}]{\zdotfill}]\
171 }
172 \begin{tabular}{rll}
173 & [\verb|\dotfill|] & [\verb|\zdotfill|]\
174 \dfptest{0.43em}
175 \dfptest{0.44em}
176 \dfptest{0.45em}
177 \dfptest{0.87em}
178 \dfptest{0.88em}
179 \dfptest{0.89em}
180 \dfptest{1.31em}
181 \dfptest{1.32em}
182 \dfptest{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 [2010/05/01 v2.17 New reference scheme for LaTeX2e (HO)]%

```

6.1.2 Load basic module

```

191 \RequirePackage{zref-base}[2010/05/01]

```

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}[2010/05/01]%
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 [2010/05/01 v2.17 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]

```

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

```
226 \def\ZREF@name{zref}
227 \ltx@ifundefined{protected}{%
228   \RequirePackage{makerobust}[2006/03/18]%
```

`\ZREF@Robust`

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

`\ZREF@Robust`

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

`\ZREF@ifDefinable`

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

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

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

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

```
245 \newif\ifZREF@found
```

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

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

6.2.4 Check for ϵ -`TeX`

The use of ϵ -`TeX` should be standard nowadays for `LATEX`. We test for ϵ -`TeX` in order to use its features later.

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

```

264 \RequirePackage{etexcmds}[2007/09/09]
265 \ifetex@unexpanded
266 \else
267   \PackageError\ZREF@name{%
268     Missing e-TeX's \string\unexpanded.\MessageBreak
269     Add \string\RequirePackage\string{etexcmds\string} before %
270     \string\documentclass%
271   }{%
272     Probably you are using some package (e.g. ConTeXt) that %
273     redefines \string\unexpanded%
274   }%
275   \expandafter\endinput
276 \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^AT_EX processes that haven't loaded this package (e.g. package xr). Therefore we provide dummy definitions.

```

277 \RequirePackage{auxhook}
278 \AddLineBeginAux{%
279   \string\providecommand\string\zref@newlabel[2]{}%
280 }

```

\ZREF@RefPrefix

```
281 \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^AT_EX's check for multiple references.
- L^AT_EX's check for changed references.

```

282 \ZREF@Robust\edef\zref@newlabel{%
283   \noexpand\@newl@bel{\ZREF@RefPrefix}%
284 }

```

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.

```

285 \ZREF@Robust\def\zref@newlist#1{%
286   \zref@iflistundefined{#1}{%
287     \@ifdefinable{Z@L@#1}{%
288       \global\expandafter\let\csname Z@L@#1\endcsname\ltx@empty
289       \PackageInfo\ZREF@name{New property list: #1}%
290     }%
291   }{%
292     \PackageError\ZREF@name{%
293       Property list '#1' already exists%
294     }\@ehc
295   }%
296 }

```

\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.

```

297 \def\zref@iflistundefined#1{%
298   \ltx@ifundefined{Z@L@#1}%
299 }

```


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

```

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

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

```

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

`\zref@listforloop`

```

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

`\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.

```

338 \ZREF@Robust\def\zref@addprop#1#2{%
339   \zref@listexists{#1}{%
340     \comma@parse{#2}{%
341       \zref@propexists\comma@entry{%
342         \zref@iflistcontainsprop{#1}\comma@entry{%
343           \PackageWarning\ZREF@name{%
344             Property ‘\comma@entry’ is already in list ‘#1’%
345           }%
346         }{%
347           \begingroup\expandafter\endgroup
348           \expandafter\g@addto@macro
349           \csname Z@L@#1\expandafter\endcsname
350           \expandafter{\csname\comma@entry\endcsname}%
351         }%
352       }%
353     }%
354   }%
355 }
```

```

352     }%
353     \ltx@gobble
354 }%
355 }%
356 }

\zref@localaddprop

357 \ZREF@Robust\def\zref@localaddprop#1#2{%
358   \zref@listexists{#1}{%
359     \comma@parse{#2}{%
360       \zref@propexists\comma@entry{%
361         \zref@iflistcontainsprop{#1}\comma@entry{%
362           \PackageWarning\ZREF@name{%
363             Property ‘\comma@entry’ is already in list ‘#1’%
364           }%
365         }{%
366           \begingroup\expandafter\endgroup
367           \expandafter\ltx@LocalAppendToMacro
368           \csname Z@L@#1\expandafter\endcsname
369           \expandafter{\csname\comma@entry\endcsname}%
370         }%
371       }%
372     \ltx@gobble
373   }%
374 }%
375 }

```

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.

```

376 \def\zref@ifpropundefined#1{%
377   \ltx@ifundefined{Z@E@#1}%
378 }

```

`\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.

```

379 \ZREF@Robust\def\zref@propexists#1{%
380   \zref@ifpropundefined{#1}{%
381     \PackageError\ZREF@name{%
382       Property ‘#1’ does not exist%
383     }\@ehc
384   }%
385 }

```

`\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.

```

386 \ZREF@Robust\def\zref@newprop{%
387   \ifstar{%
388     \let\ZREF@X\noexpand
389     \ZREF@newprop

```

```

390 }{%
391   \let\ZREF@X\ltx@empty
392   \ZREF@newprop
393 }%
394 }

```

`\ZREF@newprop`

```

395 \def\ZREF@newprop#1{%
396   \edef\ZREF@P{#1}%
397   \@onelevel@sanitize\ZREF@P
398   \begingroup
399   \ifx\ZREF@P\ZREF@par
400     \@PackageError\ZREF@name{%
401       Invalid property name ‘\ZREF@P’%
402     }{%
403       The property name ‘par’ is not allowed %
404       because of internal reasons.%
405       \MessageBreak
406       \@ehc
407     }%
408   \def\ZREF@@newprop[##1]##2{\endgroup}%
409   \else
410     \zref@ifpropundefined\ZREF@P{%
411       \endgroup
412       \PackageInfo\ZREF@name{%
413         New property: \ZREF@P
414       }%
415     }{%
416       \@PackageError\ZREF@name{%
417         Property ‘\ZREF@P’ already exists%
418       }\@ehc
419     \def\ZREF@@newprop[##1]##2{\endgroup}%
420     }%
421   \fi
422   \@ifnextchar[\ZREF@@newprop{\ZREF@@newprop[\zref@default]}%
423 }

```

`\ZREF@par`

```

424 \def\ZREF@par{par}
425 \@onelevel@sanitize\ZREF@par

```

`\ZREF@@newprop`

```

426 \def\ZREF@@newprop[#1]{%
427   \global\@namedef{Z@D@\ZREF@P}{#1}%
428   \global\expandafter\let\csname Z@X@\ZREF@P\endcsname\ZREF@X
429   \begingroup\expandafter\endgroup
430   \expandafter\ZREF@@@newprop\csname\ZREF@P\endcsname
431   \expandafter\gdef\csname Z@C@\ZREF@P\endcsname{%
432     \zref@setcurrent\ZREF@P
433 }
434 \def\ZREF@@@newprop#1{%
435   \expandafter
436   \gdef\csname Z@E@\ZREF@P\endcsname##1#1##2##3\ZREF@nil{##2}%
437 }

```

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

```

438 \ZREF@Robust\def\zref@setcurrent#1#2{%
439   \zref@propexists{#1}{%
440     \expandafter\def\csname Z@C@#1\endcsname{#2}%
441   }%
442 }

```

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

```
443 \def\ZREF@getcurrent#1{%
444   \romannumeral0%
445   \ltx@ifundefined{Z@C@#1}{%
446     \ltx@space
447   }{%
448     \expandafter\expandafter\expandafter\ltx@space
449     \csname Z@C@#1\endcsname
450   }%
451 }
```

`\ZREF@u@getcurrent`

```
452 \def\ZREF@u@getcurrent#1{%
453   \etex@unexpanded\expandafter\expandafter\expandafter{%
454     \ZREF@getcurrent{#1}%
455   }%
456 }
```

`\zref@getcurrent`

```
457 \let\zref@getcurrent\ZREF@getcurrent
```

6.2.8 Reference generation

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

```
458 \ZREF@Robust\def\zref@label#1{%
459   \zref@labelbylist{#1}\ZREF@mainlist
460 }
```

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

```
461 \ZREF@Robust\def\zref@labelbylist#1#2{%
462   \@bsphack
463   \zref@listexists{#2}{%
464     \expandafter\expandafter\expandafter\ZREF@label
465     \expandafter\expandafter\expandafter{%
466       \csname Z@L@#2\endcsname
467     }{#1}%
468   }%
469   \@esphack
470 }
```

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

```
471 \ZREF@Robust\def\zref@labelbyprops#1#2{%
472   \@bsphack
473   \begingroup
474   \toks@{}%
475   \comma@parse{#2}{%
476     \zref@ifpropundefined\comma@entry{%
477       \PackageWarning\ZREF@name{%
478         Property ‘\comma@entry’ is not known%
479       }%
480     }{%
481       \toks@\expandafter{%
482         \the\expandafter\toks@\csname\comma@entry\endcsname
483       }%
484     }%
485     \ltx@gobble
486   }%
487   \expandafter\endgroup
488   \expandafter\ZREF@label\expandafter{\the\toks@}{#1}%
489   \@esphack
490 }
```

`\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.

```
491 \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`.

```
492 \ZREF@Robust{\long\def}\zref@wrapper@immediate#1{%
493   \begingroup
494     \ZREF@immediatetrue
495     \let\ZREF@org@write\write
496     \def\write{\immediate\ZREF@org@write}%
497     #1%
498   \endgroup
499 }
```

`\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 `\ZREF@label` is made expandable in this case.

```
500 \def\ZREF@label#1#2{%
501   \if@filesw
502     \begingroup
503       \ifZREF@immediate
504         \let\ZREF@org@thepage\thepage
505       \fi
506     \protected@write\@auxout{%
507       \ifZREF@immediate
508         \let\thepage\ZREF@org@thepage
509       \fi
510     \let\ZREF@temp\ltx@empty
511     \@tfor\ZREF@P:=#1\do{%
512       \begingroup
513         \escapechar=-1 %
514         \edef\x{\endgroup
515           \def\noexpand\ZREF@P{%
516             \expandafter\string\ZREF@P
517           }%
518         }%
519       \x
520     \expandafter\ifx
521       \csname
522         \ifZREF@immediate
523           relax%
524         \else
525           Z@X@\ZREF@P%
526         \fi
527       \endcsname
528     \noexpand
529     \expandafter\let\csname Z@C@\ZREF@P\endcsname\relax
530   \fi
531   \toks@\expandafter{\ZREF@temp}%
532   \edef\ZREF@temp{%
533     \the\toks@
534     \ltx@backslashchar\ZREF@P{%
535       \expandafter\noexpand\csname Z@C@\ZREF@P\endcsname
536     }%
537   }%
538 }%
539 }{%
540   \string\zref@newlabel{#2}{\ZREF@temp}%
```

```

541     }%
542   \endgroup
543 \fi
544 }
545 \def\ZREF@addtoks#1{%
546   \toks@\expandafter\expandafter\expandafter{%
547     \expandafter\the\expandafter\toks@#1%
548   }%
549 }

```

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^AT_EX 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 pdf_TE_X 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.

```

550 \def\zref@ifrefundefined#1{%
551   \ltx@ifundefined{Z@R@#1}%
552 }

```

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

```

553 \ZREF@IfDefinable\zifrefundefined\def{%
554   #1{%
555     \zref@wrapper@babel\ZREF@ifrefundefined{#1}%
556   }%
557 }

```

`\ZREF@ifrefundefined`

```

558 \def\ZREF@ifrefundefined#1{%
559   \zref@refused{#1}%
560   \zref@ifrefundefined{#1}%
561 }

```

`\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^AT_EX and a warning is given.

```

562 \ZREF@Robust\def\zref@refused#1{%
563   \zref@wrapper@babel\ZREF@refused{#1}%
564 }

```

`\ZREF@refused`

```

565 \def\ZREF@refused#1{%
566   \zref@ifrefundefined{#1}{%
567     \protect\G@refundefinedtrue
568     \@latex@warning{%
569       Reference ‘#1’ on page \thepage \space undefined%
570     }%
571   }{}%
572 }

```

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

```

573 \def\zref@ifrefcontainsprop#1#2{%
574   \zref@ifrefundefined{#1}{%
575     \ltx@secondoftwo
576   }{%
577     \expandafter\ZREF@ifrefcontainsprop
578     \csname Z@E@#2\expandafter\endcsname
579     \csname#2\expandafter\expandafter\expandafter\endcsname
580     \expandafter\expandafter\expandafter{%
581       \csname Z@R@#1\endcsname
582     }%
583   }%
584 }
585 \def\ZREF@ifrefcontainsprop#1#2#3{%
586   \expandafter\ifx\expandafter\ZREF@novalue
587   #1#3#2\ZREF@novalue\ZREF@nil\ltx@empty
588   \expandafter\ltx@secondoftwo
589   \else
590     \expandafter\ltx@firstoftwo
591   \fi
592 }
593 \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.

```

594 \def\ZREF@extract#1#2{%
595   \romannumeral0%
596   \ltx@ifundefined{Z@D@#2}{%
597     \expandafter\ltx@space\zref@default
598   }{%
599     \expandafter\expandafter\expandafter\ZREF@@extract
600     \expandafter\expandafter\expandafter{%
601       \csname Z@D@#2\endcsname
602     }{#1}{#2}%
603   }%
604 }

```

`\ZREF@@extract`

```

605 \def\ZREF@@extract#1#2#3{%
606   \expandafter\expandafter\expandafter\ltx@space
607   \zref@extractdefault{#2}{#3}{#1}%
608 }

```

`\ZREF@wu@extract`

```

609 \def\ZREF@wu@extract#1#2{%
610   \etex@unexpanded\expandafter\expandafter\expandafter{%
611     \ZREF@extract{#1}{#2}%
612   }%
613 }

```

`\zref@extract`

```

614 \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.

```

615 \def\ZREF@extractdefault#1#2#3{%
616   \romannumeral0%
617   \zref@ifrefundefined{#1}\ltx@firstoftwo{%
618     \zref@ifpropundefined{#2}\ltx@firstoftwo\ltx@secondoftwo
619   }{%

```

```

620 \ltx@space
621 #3%
622 }{%
623 \expandafter\expandafter\expandafter\ltx@space
624 \csname Z@E@#2\expandafter\expandafter\expandafter\endcsname
625 \csname Z@R@#1\expandafter\endcsname
626 \csname#2\endcsname{#3}\ZREF@nil
627 }%
628 }

```

`\ZREF@wu@extractdefault`

```

629 \def\ZREF@wu@extractdefault#1#2#3{%
630 \etex@unexpanded\expandafter\expandafter\expandafter{%
631 \ZREF@extractdefault{#1}{#2}{#3}%
632 }%
633 }

```

`\zref@extractdefault`

```

634 \let\zref@extractdefault\ZREF@extractdefault

```

`\ZREF@wrapper@unexpanded`

```

635 \ZREF@Robust{\long\def}\ZREF@wrapper@unexpanded#1{%
636 \let\zref@wrapper@unexpanded\ltx@firstofone
637 \let\zref@getcurrent\ZREF@wu@getcurrent
638 \let\zref@extractdefault\ZREF@wu@extractdefault
639 \let\zref@extract\ZREF@wu@extract
640 #1%
641 \let\zref@wrapper@unexpanded\ZREF@wrapper@unexpanded
642 \let\zref@getcurrent\ZREF@getcurrent
643 \let\zref@extractdefault\ZREF@extractdefault
644 \let\zref@extract\ZREF@extract
645 }

```

`\zref@wrapper@unexpanded`

```

646 \ltx@ifundefined{etex@unexpanded}{%
647 \let\zref@wrapper@unexpanded\ltx@firstofone
648 }{%
649 \let\zref@wrapper@unexpanded\ZREF@wrapper@unexpanded
650 }

```

6.2.10 Compatibility with babel

`\zref@wrapper@babel`

```

651 \ZREF@Robust{\long\def}\zref@wrapper@babel#1#2{%
652 \ifcsname if@safe@actives\endcsname
653 \expandafter\ltx@firstofone
654 \else
655 \expandafter\ltx@secondoftwo
656 \fi
657 {%
658 \if@safe@actives
659 \expandafter\ltx@secondoftwo
660 \else
661 \expandafter\ltx@firstoftwo
662 \fi
663 {%
664 \begingroup
665 \csname @safe@activestrue\endcsname
666 \edef\x{#2}%
667 \expandafter\endgroup
668 \expandafter\ZREF@wrapper@babel\expandafter{\x}{#1}%

```



```

669   }%
670  }{%
671   #1{#2}%
672  }%
673 }
674 \long\def\ZREF@wrapper@babel#1#2{%
675   #2{#1}%
676 }

```

6.2.11 Unique counter support

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

```

677 \ZREF@Robust\def\zref@require@unique{%
678   \@ifundefined{c@zref@unique}{%
679     \begingroup
680     \let\@addtoreset\ltx@gobbletwo
681     \newcounter{zref@unique}%
682     \endgroup

```

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

```

683   \renewcommand*\thezref@unique{%
684     zref@number\c@zref@unique
685   }%
686 }{%
687 }

```

6.2.12 Utilities

`\ZREF@number`

```

688 \ltx@ifundefined{numexpr}{%
689   \let\ZREF@number\number
690 }{%
691   \def\ZREF@number#1{\the\numexpr#1}%
692 }

```

6.2.13 Setup

`\zref@setdefault` Standard L^AT_EX 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`.

```

693 \ZREF@Robust\def\zref@setdefault#1{%
694   \def\zref@default{#1}%
695 }

```

`\zref@default` Now we initialize `\zref@default` with the same value that L^AT_EX uses for its undefined references.

```

696 \zref@setdefault{%
697   \nfss@text{\reset@font\bfseries ??}%
698 }

```

Main property list.

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

```

699 \ZREF@Robust\def\zref@setmainlist#1{%
700   \def\ZREF@mainlist{#1}%
701 }
702 \zref@setmainlist{main}

```

Now we create the list.

```

703 \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^AT_EX 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

```
704 \zref@newprop{default}{\@currentlabel}
705 \zref@newprop*{page}{\thepage}
706 \zref@addprop\ZREF@mainlist{default,page}
```

Mark successful loading

```
707 \let\ZREF@base@ok=Y
708 </base>
```

6.3 Module user

```
709 <*user>
710 \NeedsTeXFormat{LaTeX2e}
711 \ProvidesPackage{zref-user}%
712 [2010/05/01 v2.17 Module user for zref (HO)]%
713 \RequirePackage{zref-base}[2010/05/01]
714 \ifx\ZREF@base@ok Y%
715 \else
716 \expandafter\endinput
717 \fi
```

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

First we define the pendants to the standard L^AT_EX 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.

```
718 \newcommand*\zlabel{%
719 \ifx\label\ltx@gobble
720 \expandafter\ltx@gobble
721 \else
722 \expandafter\zref@wrapper@babel\expandafter\zref@label
723 \fi
724 }%
```

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

```
725 \newcommand*\zref}[2][default]{% robust because of optional argument
726 \zref@propexists{#1}{%
727 \zref@wrapper@babel\ZREF@zref{#2}{#1}%
728 }%
729 }%
730 \def\ZREF@zref#1{%
731 \zref@refused{#1}%
732 \zref@extract{#1}%
733 }%
```

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

```
734 \ZREF@IfDefinable\zpageref\def{%
735 {\zref[page]}%
736 }
```

`\zrefused` For the following expandible user macros `\zrefused` should be used to notify L^AT_EX in case of undefined references.

```
737 \ZREF@ifDefinable\zrefused\def{%
738   {\zref@refused}%
739 }

740 </user>
```

6.4 Module `abspage`

```
741 <*abspage>
742 \NeedsTeXFormat{LaTeX2e}
743 \ProvidesPackage{zref-abspage}%
744   [2010/05/01 v2.17 Module abspage for zref (HO)]%
745 \RequirePackage{zref-base}[2010/05/01]
746 \ifx\ZREF@base@ok Y%
747 \else
748   \expandafter\endinput
749 \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`.

```
750 \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_EX files.

```
751 \begingroup
752   \let\@addtoreset\ltx@gobbletwo
753   \newcounter{abspage}%
754 \endgroup
755 \setcounter{abspage}{0}%
756 \AtBeginShipout{%
757   \stepcounter{abspage}%
758 }%
759 \zref@newprop*{abspage}[0]{\the\c@abspage}%
760 \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.

```
761 </abspage>
```

6.5 Module `counter`

```
762 <*counter>
763 \NeedsTeXFormat{LaTeX2e}
764 \ProvidesPackage{zref-counter}%
765   [2010/05/01 v2.17 Module counter for zref (HO)]%
766 \RequirePackage{zref-base}[2010/05/01]
767 \ifx\ZREF@base@ok Y%
768 \else
769   \expandafter\endinput
770 \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.

```
771 \zref@newprop{counter}{%
772 \zref@addprop\ZREF@mainlist{counter}}
```

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

```
773 \AtBeginDocument{%
774   \ZREF@patch{refstepcounter}{%
775     \def\refstepcounter#1{%
```

```

776     \zref@setcurrent{counter}{#1}%
777     \ZREF@org@refstepcounter{#1}%
778   }%
779 }%
780 }
781 </counter>

```

6.6 Module lastpage

```

782 <*lastpage>
783 \NeedsTeXFormat{LaTeX2e}
784 \ProvidesPackage{zref-lastpage}%
785 [2010/05/01 v2.17 Module lastpage for zref (HO)]%
786 \RequirePackage{zref-base}[2010/05/01]
787 \RequirePackage{zref-abspage}[2010/05/01]
788 \RequirePackage{atveryend}[2009/12/07]
789 \ifx\ZREF@base@ok Y%
790 \else
791   \expandafter\endinput
792 \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.

```

793 \zref@newlist{LastPage}
794 \AfterLastShipout{%
795   \if@filesw
796     \begingroup
797       \advance\c@page\m@ne
798       \toks@\expandafter\expandafter\expandafter{%
799         \expandafter\Z@L@main
800         \Z@L@LastPage
801       }%
802       \expandafter\zref@wrapper@immediate\expandafter{%
803         \expandafter\ZREF@label\expandafter{\the\toks@}{LastPage}%
804       }%
805     \endgroup
806 \fi
807 }

```

`\zref@iflastpage`

```

808 \def\zref@iflastpage#1{%
809   \ifnum\zref@extractdefault{#1}{abspage}{-1}=%
810     \zref@extractdefault{LastPage}{abspage}{-2} %
811     \expandafter\ltx@firstoftwo
812   \else
813     \expandafter\ltx@secondoftwo
814   \fi
815 }

```

`\ziflastpage`

```

816 \ZREF@IfDefinable\ziflastpage\def{%
817   {\zref@wrapper@babel\ZREF@iflastpage}%
818 }

```

`ZREF@iflastpage`

```

819 \def\ZREF@iflastpage#1{%
820   \zref@refused{LastPage}%
821   \zref@refused{#1}%
822   \zref@iflastpage{#1}%
823 }

```

```

824 </lastpage>

```

6.7 Module thepage

```
825 <*thepage>
826 \NeedsTeXFormat{LaTeX2e}
827 \ProvidesPackage{zref-thepage}%
828 [2010/05/01 v2.17 Module thepage for zref (HO)]%
829 \RequirePackage{zref-base}[2010/05/01]
830 \ifx\ZREF@base@ok Y%
831 \else
832 \expandafter\endinput
833 \fi

834 \RequirePackage{atbegshi}
835 \RequirePackage{zref-abspage}[2010/05/01]

836 \zref@newlist{thepage}
837 \zref@addprop{thepage}{page}
838 \AtBeginShipout{%
839 \zref@wrapper@immediate{%
840 \zref@labelbylist{thepage\the\value{abspage}}{thepage}%
841 }%
842 }
```

\zref@thepage@name

```
843 \ltx@ifundefined{numexpr}{%
844 \def\zref@thepage@name#1{thepage\number#1}%
845 }{%
846 \def\zref@thepage@name#1{thepage\the\numexpr#1}%
847 }
```

\zref@thepage

```
848 \def\zref@thepage#1{%
849 \zref@extract{\zref@thepage@name{#1}}{page}%
850 }%
```

\zref@thepage@refused

```
851 \ZREF@Robust\def\zref@thepage@refused#1{%
852 \zref@refused{\zref@thepage@name{#1}}%
853 }%
```

\zthepage

```
854 \ZREF@ifdefinable\zthepage\def{%
855 #1{%
856 \zref@thepage@refused{#1}%
857 \zref@thepage{#1}%
858 }%
859 }
```

```
860 </thepage>
```

6.8 Module nextpage

```
861 <*nextpage>
862 \NeedsTeXFormat{LaTeX2e}
863 \ProvidesPackage{zref-nextpage}%
864 [2010/05/01 v2.17 Module nextpage for zref (HO)]%
865 \RequirePackage{zref-base}[2010/05/01]
866 \ifx\ZREF@base@ok Y%
867 \else
868 \expandafter\endinput
869 \fi

870 \RequirePackage{zref-abspage}[2010/05/01]
871 \RequirePackage{zref-thepage}[2010/05/01]
```

```

872 \RequirePackage{zref-lastpage}[2010/05/01]
873 \RequirePackage{uniquecounter}[2009/12/18]

874 \UniqueCounterNew{znextpage}
875
876 \newcommand*{\znextpagesetup}{%
877   \afterassignment\ZREF@np@setup@i
878   \def\ZREF@np@call@unknown##1%
879 }
880 \def\ZREF@np@setup@i{%
881   \afterassignment\ZREF@np@setup@ii
882   \def\ZREF@np@call@nonext##1%
883 }
884 \def\ZREF@np@setup@ii{%
885   \def\ZREF@np@call@next##1%
886 }
887 \def\ZREF@np@call@unknown#1{#1}
888 \def\ZREF@np@call@nonext#1{#1}
889 \def\ZREF@np@call@next#1{#1}
890 \ZREF@IfDefinable\znextpage\def{%
891   {\UniqueCounterCall{znextpage}{\ZREF@nextpage}}%
892 }%
893 \newcommand*{\znonextpagename}{}
894 \newcommand*{\zunknownnextpagename}{\Z@D@page}
895 \def\ZREF@nextpage#1{%
896   \begingroup
897     \def\ZREF@refname@this{zref@np#1}%
898     \zref@labelbyprops\ZREF@refname@this{abspage}%
899     \chardef\ZREF@call=0 % unknown
900     \ZREF@ifrefundefined\ZREF@refname@this{%
901       }{%
902         \edef\ZREF@pagenum@this{%
903           \zref@extractdefault\ZREF@refname@this{abspage}{0}%
904         }%
905         \edef\ZREF@refname@next{%
906           \zref@thepage@name{%
907             \the\numexpr\ZREF@pagenum@this+1%
908           }%
909         }%
910         \ifnum\ZREF@pagenum@this>0 %
911           \ZREF@ifrefundefined{LastPage}{%
912             \zref@ifrefundefined\ZREF@refname@next{%
913               }{%
914                 \chardef\ZREF@call=2 % next page
915               }%
916             }{%
917               \edef\ZREF@pagenum@last{%
918                 \zref@extractdefault{LastPage}{abspage}{0}%
919               }%
920               \ifnum\ZREF@pagenum@this<\ZREF@pagenum@last\ltx@space
921                 \ZREF@ifrefundefined\ZREF@refname@next{%
922                   }{%
923                     \chardef\ZREF@call=2 % next page
924                   }%
925                 }%
926                 \else
927                   \ifnum\ZREF@pagenum@this=\ZREF@pagenum@this\ltx@space
928                     \chardef\ZREF@call=1 % no next page
929                   \fi
930                 }%
931               }%
932             }%
933           \edef\x{%

```

```

934     \endgroup
935     \ifcase\ZREF@call
936         \noexpand\ZREF@np@call@unknown{%
937             \noexpand\zunknownnextpagename
938         }%
939     \or
940         \noexpand\ZREF@np@call@nonext{%
941             \noexpand\znonextpagename
942         }%
943     \else
944         \noexpand\ZREF@np@call@next{%
945             \noexpand\zref@extract{\ZREF@refname@next}{page}%
946         }%
947     \fi
948 }%
949 \x
950 }
951 </nextpage>

```

6.9 Module `totpages`

```

952 <*totpages>
953 \NeedsTeXFormat{LaTeX2e}
954 \ProvidesPackage{zref-totpages}%
955 [2010/05/01 v2.17 Module totpages for zref (HO)]%
956 \RequirePackage{zref-base}[2010/05/01]
957 \ifx\ZREF@base@ok Y%
958 \else
959     \expandafter\endinput
960 \fi

```

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

```

961 \RequirePackage{zref-abspage}[2010/05/01]
962 \RequirePackage{zref-lastpage}[2010/05/01]

```

`\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.

```

963 \newcommand*{\ztotpages}{%
964     \zref@extractdefault{LastPage}{abspage}{0}%
965 }

```

Also we mark the reference `LastPage` as used:

```

966 \AtBeginDocument{%
967     \zref@refused{LastPage}%
968 }
969 </totpages>

```

6.10 Module `pagelayout`

```

970 <*pagelayout>
971 \NeedsTeXFormat{LaTeX2e}
972 \ProvidesPackage{zref-pagelayout}%
973 [2010/05/01 v2.17 Module pagelayout for zref (HO)]%
974 \RequirePackage{zref-base}[2010/05/01]
975 \ifx\ZREF@base@ok Y%
976 \else
977     \expandafter\endinput
978 \fi
979 \RequirePackage{zref-thepage}[2010/05/01]
980 \RequirePackage{ifluatex}[2010/03/01]
981 \RequirePackage{atveryend}[2010/03/24]
982 \ifluatex

```

```

983 \ifnum\luatexversion<39 %
984 \else
985   \begingroup
986     \escapechar=-1 %
987     \def\ZREF@temp#1{%
988       \ltx@ifundefined{\string#1}{%
989         \let#1\ltx@undefined
990         \directlua{%
991           if tex.enableprimitives then %
992             tex.enableprimitives('', {\string#1})%
993           end%
994         }%
995       \ltx@ifundefined{\string#1}{%
996         }{%
997           \global#1=#1%
998           \@PackageInfoNoLine{zref-pagelayout}{%
999             \string#1 enabled%
1000           }%
1001         }%
1002       }{%
1003     }%
1004     \ZREF@temp\pdfpagewidth
1005     \ZREF@temp\pdfpageheight
1006     \ZREF@temp\pdfhorigin
1007     \ZREF@temp\pdfvorigin
1008   \endgroup
1009 \fi
1010 \fi

1011 \def\ZREF@temp#1{%
1012   \begingroup
1013     \escapechar=-1 %
1014   \ltx@ifundefined{\string#1}{\endgroup}{%
1015     \edef\x{%
1016       \endgroup
1017       \noexpand\zref@newprop*{\string#1}%
1018         [\number\noexpand#1]% hash-ok
1019         {\number\noexpand#1}%
1020       \noexpand\zref@addprop{thepage}{\string#1}%
1021     }%
1022     \x
1023   }%
1024 }

1025 \ZREF@temp\mag
1026 \ZREF@temp\paperwidth
1027 \ZREF@temp\paperheight
1028 \ZREF@temp\stockwidth
1029 \ZREF@temp\stockheight
1030 \ZREF@temp\pdfpagewidth
1031 \ZREF@temp\pdfpageheight
1032 \ZREF@temp\pdfhorigin
1033 \ZREF@temp\pdfvorigin
1034 \ZREF@temp\hoffset
1035 \ZREF@temp\voffset
1036 \ZREF@temp\topmargin
1037 \ZREF@temp\oddsidemargin
1038 \ZREF@temp\evensidemargin
1039 \ZREF@temp\textwidth
1040 \ZREF@temp\textheight
1041 \ZREF@temp\headheight
1042 \ZREF@temp\headsep
1043 \ZREF@temp\footskip
1044 \ZREF@temp\marginparwidth

```



```

1045 \ZREF@temp\marginparsep
1046 \ZREF@temp\columnwidth
1047 \ZREF@temp\columnsep

```

```
\ifZREF@pl@list
```

```
1048 \ltx@newif\ifZREF@pl@list
```

```
\zref@listpagelayout
```

```

1049 \ZREF@IfDefinable\zlistpagelayout\def{%
1050   {\ZREF@pl@listtrue}%
1051 }

```

```
\ZREF@pl@AfterLastShipout
```

```

1052 \def\ZREF@pl@AfterLastShipout{%
1053   \ifZREF@pl@list
1054     \edef\ZREF@page@max{\the\value{abspage}}%
1055     \ltx@ifundefined{ZREF@org@testdef}{%
1056       \let\ZREF@org@testdef\@testdef
1057       \def\@testdef##1##2##3{%
1058         \ZREF@org@testdef{##1}{##2}{##3}%
1059         \def\ZREF@temp{##1}%
1060         \ifx\ZREF@temp\ZREF@RefPrefix
1061           \expandafter\xdef\csname##1@##2\endcsname{##3}%
1062         \fi
1063       }%
1064     }{}%
1065     \AtVeryEndDocument{\ZREF@pl@AtVeryEnd}%
1066   \fi
1067 }

```

```
\ZREF@pl@AtVeryEnd
```

```

1068 \def\ZREF@pl@AtVeryEnd{%
1069   \begingroup
1070   \toks@{Page layout parameters:\MessageBreak}%
1071   \count@=1 %
1072   \ZREF@pl@ListPage
1073   \edef\x{\endgroup
1074     \noexpand\@PackageInfoNoLine{zref-pagelayout}{\the\toks@}%
1075   }%
1076   \x
1077 }

```

```
\ZREF@pl@ListPage
```

```

1078 \def\ZREF@pl@ListPage{%
1079   \edef\x{%
1080     \toks@=%%
1081     \the\toks@
1082     Page \the\count@:\noexpand\MessageBreak
1083     \zref@ifrefundefined{thepage\the\count@}{-}{%
1084       \ltx@space\ltx@space mag = %
1085       \zref@extract{thepage\the\count@}{mag}%
1086       \noexpand\MessageBreak
1087       \ZREF@pl@ListEntry{paperwidth}%
1088       \ZREF@pl@ListEntry{paperheight}%
1089       \ZREF@pl@ListEntry{stockwidth}%
1090       \ZREF@pl@ListEntry{stockheight}%
1091       \ZREF@pl@ListEntry{pdfpagewidth}%
1092       \ZREF@pl@ListEntry{pdfpageheight}%
1093       \ZREF@pl@ListEntry{pdfhorigin}%
1094       \ZREF@pl@ListEntry{pdfvorigin}%
1095       \ZREF@pl@ListEntry{hoffset}%
1096       \ZREF@pl@ListEntry{voffset}%

```

```

1097     \ZREF@pl@ListEntry{topmargin}%
1098     \ZREF@pl@ListEntry{oddsidemargin}%
1099     \ZREF@pl@ListEntry{evensidemargin}%
1100     \ZREF@pl@ListEntry{textwidth}%
1101     \ZREF@pl@ListEntry{textheight}%
1102     \ZREF@pl@ListEntry{headheight}%
1103     \ZREF@pl@ListEntry{headsep}%
1104     \ZREF@pl@ListEntry{footskip}%
1105     \ZREF@pl@ListEntry{marginparwidth}%
1106     \ZREF@pl@ListEntry{marginparsep}%
1107     \ZREF@pl@ListEntry{columnwidth}%
1108     \ZREF@pl@ListEntry{columnsep}%
1109   }%
1110 }%
1111 }\x
1112 \ifnum\ZREF@page@max>\count@
1113   \advance\count@ by\ltx@one
1114 \else
1115   \expandafter\ltx@gobble
1116 \fi
1117 \ZREF@pl@ListPage
1118 }

```

\ZREF@pl@ListEntry

```

1119 \def\ZREF@pl@ListEntry#1{%
1120   \zref@ifpropundefined{#1}{%
1121     }{%
1122     \zref@ifrefcontainsprop{thepage\the\count@}{#1}{%
1123       \ltx@space\ltx@space#1 = %
1124       \zref@extract{thepage\the\count@}{#1}sp = %
1125       \the\dimexpr\zref@extract{thepage\the\count@}{#1}sp\relax
1126       \noexpand\MessageBreak
1127     }{%}
1128   }%
1129 }

1130 \AfterLastShipout{%
1131   \ZREF@pl@AfterLastShipout
1132 }

1133 </pagelayout>

```

6.11 Module pageattr

```

1134 <*pageattr>
1135 \NeedsTeXFormat{LaTeX2e}
1136 \ProvidesPackage{zref-pageattr}%
1137 [2010/05/01 v2.17 Module pageattr for zref (HO)]%
1138 \RequirePackage{zref-base}[2010/05/01]
1139 \ifx\ZREF@base@ok Y%
1140 \else
1141   \expandafter\endinput
1142 \fi

1143 \RequirePackage{ifluatex}[2010/03/01]

1144 \ifluatex
1145   \ifnum\luatexversion<39 %
1146   \else
1147     \begingroup
1148     \escapechar=-1 %
1149     \def\ZREF@temp#1{%
1150       \ltx@ifundefined{string#1}{%
1151         \let#1\ltx@undefined

```

```

1152     \directlua{%
1153         if tex.enableprimitives then %
1154             tex.enableprimitives('', {'\string#1'})%
1155         end%
1156     }%
1157     \ltx@ifundefined{\string#1}{%
1158     }{%
1159         \global#1=#1%
1160         \@PackageInfoNoLine{zref-pageattr}{%
1161             \string#1 enabled%
1162         }%
1163     }%
1164 }{%
1165 }%
1166     \ZREF@temp\pdfpageattr
1167     \ZREF@temp\pdfpagesattr
1168 \endgroup
1169 \fi
1170 \fi

1171 \let\ZREF@temp=N%
1172 \ltx@ifundefined{pdfpageattr}{%
1173     \@PackageInfoNoLine{zref-pageattr}{%
1174         \string\pdfpageattr\space is not available%
1175     }%
1176     \def\zref@pdfpageattr#1{%
1177     \def\zref@pdfpageattr@used#1{%
1178 }{%
1179     \RequirePackage{zref-thepage}[2010/05/01]%
1180     \zref@newprop*{pdfpageattr}[]{\zref@hex{\the\pdfpageattr}}%
1181     \zref@addprop{thepage}{pdfpageattr}%
1182     \let\ZREF@temp=Y%
1183 }
1184 \ltx@ifundefined{pdfpagesattr}{%
1185     \@PackageInfoNoLine{zref-pageattr}{%
1186         \string\pdfpagesattr\space is not available%
1187     }%
1188     \def\zref@pdfpagesattr{%
1189     \def\zref@pdfpagesattr@used{%
1190 }{%
1191     \RequirePackage{zref-lastpage}[2010/05/01]%
1192     \zref@newprop*{pdfpagesattr}[]{\zref@hex{\the\pdfpagesattr}}%
1193     \zref@addprop{LastPage}{pdfpagesattr}%
1194     \let\ZREF@temp=Y%
1195 }%

1196 \ifx\ZREF@temp N%
1197     \expandafter\endinput
1198 \fi

1199 \RequirePackage{zref-abspage}[2010/05/01]
1200 \RequirePackage{atveryend}[2010/03/24]
1201 \RequirePackage{pdftexcmds}[2010/04/01]
1202 \let\ZREF@temp=Y%
1203 \ltx@ifundefined{pdf@escapehex}{\let\ZREF@temp=N}{%
1204 \ltx@ifundefined{pdf@unescapehex}{\let\ZREF@temp=N}{%
1205 \ifx\ZREF@temp N%
1206     \let\zref@hex\ltx@firstofone
1207     \let\zref@unhex\ltx@firstofone
1208 \else
1209     \let\zref@hex\pdf@escapehex
1210     \let\zref@unhex\pdf@unescapehex
1211 \fi

```

\ifZREF@pa@list

```
1212 \ltx@newif\ifZREF@pa@list
```

```
\zref@listpageattr
```

```
1213 \ZREF@IfDefinable\zlistpageattr\def{%  
1214   {\ZREF@pa@listtrue}%  
1215 }
```

```
\ZREF@pa@AfterLastShipout
```

```
1216 \def\ZREF@pa@AfterLastShipout{%  
1217   \ifZREF@pa@list  
1218     \edef\ZREF@page@max{\the\value{abspage}}%  
1219     \ltx@ifundefined{ZREF@org@testdef}{%  
1220       \let\ZREF@org@testdef\@testdef  
1221       \def\@testdef##1##2##3{%  
1222         \ZREF@org@testdef{##1}{##2}{##3}%  
1223         \def\ZREF@temp{##1}%  
1224         \ifx\ZREF@temp\ZREF@RefPrefix  
1225           \expandafter\xdef\csname##1@##2\endcsname{##3}%  
1226         \fi  
1227       }%  
1228     }{}%  
1229     \AtVeryEndDocument{\ZREF@pa@AtVeryEnd}%  
1230   \fi  
1231 }
```

```
\ZREF@pa@AtVeryEnd
```

```
1232 \ltx@ifundefined{pdfpageattr}{%  
1233   \def\ZREF@pa@AtVeryEnd{}%  
1234 }{%  
1235   \def\ZREF@pa@AtVeryEnd{%  
1236     \begingroup  
1237       \toks@{List of \ltx@backslashchar pdfpageattr:\MessageBreak}%  
1238       \count@=1 %  
1239       \ZREF@pa@ListPage  
1240       \edef\x{\endgroup  
1241         \noexpand\@PackageInfoNoLine{zref-pageattr}{%  
1242           \the\toks@  
1243         }%  
1244       }%  
1245     \x  
1246   }%
```

```
\zref@pageattr
```

```
1247 \def\zref@pageattr#1{%  
1248   \zref@unhex{%  
1249     \zref@extract{thepage\ZREF@number#1}{pdfpageattr}%  
1250   }%  
1251 }
```

```
\zref@pageattr@used
```

```
1252 \ZREF@Robust\def\zref@pageattr@used#1{%  
1253   \zref@refused{thepage\ZREF@number#1}%  
1254 }
```

```
\ZREF@pa@ListPage
```

```
1255 \def\ZREF@pa@ListPage{%  
1256   \edef\x{%  
1257     \toks@={%  
1258       \the\toks@  
1259       Page \the\count@:%  
1260     \noexpand\MessageBreak
```

```

1261     \zref@ifrefundefined{thepage\the\count@}{}{%
1262     <<\zref@pdfpageattr\count@>>%
1263     \noexpand\MessageBreak
1264     }%
1265   }%
1266 } \x
1267 \ifnum\ZREF@page@max>\count@
1268   \advance\count@ by\ltx@one
1269   \else
1270     \expandafter\ltx@gobble
1271   \fi
1272   \ZREF@pa@ListPage
1273 }%
1274 }

1275 \ltx@ifundefined{pdfpagesattr}{%
1276 }{%

```

\zref@pdfpagesattr

```

1277 \def\zref@pdfpagesattr{%
1278   \zref@unhex{%
1279     \zref@extract{LastPage}{pdfpagesattr}%
1280   }%
1281 }%

```

\zref@pdfpagesattr@used

```

1282 \ZREF@Robust\def\zref@pdfpagesattr@used{%
1283   \zref@refused{LastPage}%
1284 }%

1285 \ltx@LocalAppendToMacro\ZREF@pa@AtVeryEnd{%
1286   \@PackageInfoNoLine{zref-pageattr}{%
1287     \ltx@backslashchar pdfpagesattr:\MessageBreak
1288     <<\zref@pdfpagesattr>>%
1289     \MessageBreak
1290   }%
1291 }%
1292 }
1293 \AfterLastShipout{%
1294   \ZREF@pa@AfterLastShipout
1295 }
1296 </pageattr>

```

6.12 Module marks

```

1297 <*marks>
1298 \NeedsTeXFormat{LaTeX2e}
1299 \ProvidesPackage{zref-marks}%
1300 [2010/05/01 v2.17 Module marks for zref (HO)]%
1301 \RequirePackage{zref-base}[2010/05/01]
1302 \ifx\ZREF@base@ok Y%
1303 \else
1304   \expandafter\endinput
1305 \fi

1306 \RequirePackage{kvsetkeys}[2009/07/30]
1307 \newcommand*{\zref@marks@register}[3][ ]{%
1308   \edef\ZREF@TempName{#1}%
1309   \edef\ZREF@TempNum{\ZREF@number#2}%
1310   \ifnum\ZREF@TempNum<\ltx@zero %
1311     \PackageError\ZREF@name{%
1312       \string\zref@marks@register\ltx@space is called with invalid%
1313       \MessageBreak

```

```

1314     marks register number (\ZREF@TempNum)%
1315 }{%
1316     Use '0' or the command, defined by \string\newmarks.\MessageBreak
1317     \@ehc
1318 }%
1319 \else
1320     \ifx\ZREF@TempName\ltx@empty
1321         \edef\ZREF@TempName{mark\romannumeral\ZREF@TempNum}%
1322     \else
1323         \edef\ZREF@TempName{marks\ZREF@TempName}%
1324     \fi
1325     \ZREF@MARKS@DefineProp{top}%
1326     \ZREF@MARKS@DefineProp{first}%
1327     \ZREF@MARKS@DefineProp{bot}%
1328     \kv@parse{#3}{%
1329         \ifx\kv@value\relax
1330             \def\kv@value{top,first,bot}%
1331         \fi
1332         \edef\ZREF@temp{\expandafter\ltx@car\kv@key X\@nil}%
1333         \ifx\ZREF@temp\ZREF@STAR
1334             \edef\kv@key{\expandafter\ltx@cdr\kv@key\@nil}%
1335             \zref@newlist\kv@key
1336         \fi
1337         \expandafter\comma@parse\expandafter{\kv@value}{%
1338             \ifcase0\ifx\comma@entry\ZREF@NAME@top 1\else
1339                 \ifx\comma@entry\ZREF@NAME@first 1\else
1340                     \ifx\comma@entry\ZREF@NAME@bot 1\fi\fi\fi\ltx@space
1341             \PackageWarning{zref-marks}{%
1342                 Use 'top', 'first' or 'bot' for the list values%
1343                 \MessageBreak
1344                 in the third argument of \string\zref@marks@register.%
1345                 \MessageBreak
1346                 Ignoring unkwon value '\comma@entry'%
1347             }%
1348         \else
1349             \zref@addprop{\kv@key}{\comma@entry\ZREF@TempName}%
1350         \fi
1351         \ltx@gobble
1352     }%
1353     \ltx@gobbletwo
1354 }%
1355 \fi
1356 }
1357 \def\ZREF@STAR{*}
1358 \def\ZREF@NAME@top{top}
1359 \def\ZREF@NAME@first{first}
1360 \def\ZREF@NAME@bot{bot}
1361 \def\ZREF@MARKS@DefineProp#1{%
1362     \zref@ifpropundefined{#1\ZREF@TempName}{%
1363         \ifnum\ZREF@TempNum=\ltx@zero
1364             \begingroup
1365                 \edef\x{\endgroup
1366                     \noexpand\zref@newprop*{#1\ZREF@TempName} [] {%
1367                         \expandafter\noexpand\csname#1mark\endcsname
1368                     }%
1369                 }%
1370             \x
1371         \else
1372             \begingroup
1373                 \edef\x{\endgroup
1374                     \noexpand\zref@newprop*{#1\ZREF@TempName} [] {%
1375                         \expandafter\noexpand\csname#1marks\endcsname

```

```

1376         \ZREF@TempNum
1377     }%
1378 }%
1379 \x
1380 \fi
1381 }{%
1382 \PackageWarning{zref-marks}{%
1383 \string\zref@marks@register\ltx@space does not generate the%
1384 \MessageBreak
1385 new property '#1\ZREF@TempName', because\MessageBreak
1386 it is already defined%
1387 }%
1388 }%
1389 }
1390 </marks>

```

6.13 Module runs

This module does not use the label-reference-system. The reference changes with each L^AT_EX run and would force a rerun warning always.

```

1391 <*runs>
1392 \NeedsTeXFormat{LaTeX2e}
1393 \ProvidesPackage{zref-runs}%
1394 [2010/05/01 v2.17 Module runs for zref (HO)]%

```

\zruns

```

1395 \providecommand*\zruns{0}%
1396 \AtBeginDocument{%
1397 \edef\zruns{\number\numexpr\zruns+1}%
1398 \begingroup
1399 \def\on@line{}%
1400 \PackageInfo{zref-runs}{LaTeX runs: \zruns}%
1401 \if@filesw
1402 \immediate\write\@mainaux{%
1403 \string\gdef\string\zruns{\zruns}%
1404 }%
1405 \fi
1406 \endgroup
1407 }
1408 </runs>

```

6.14 Module perpage

```

1409 <*perpage>
1410 \NeedsTeXFormat{LaTeX2e}
1411 \ProvidesPackage{zref-perpage}%
1412 [2010/05/01 v2.17 Module perpage for zref (HO)]%
1413 \RequirePackage{zref-base}[2010/05/01]
1414 \ifx\ZREF@base@ok Y%
1415 \else
1416 \expandafter\endinput
1417 \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.

```
1418 \RequirePackage{zref-abspage}[2010/05/01]
```

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

```
1419 \zref@newprop*{pagevalue}[0]{\number\c@page}
```

```
1420 \zref@newlist{perpage}
1421 \zref@addprop{perpage}{abspage,page,pagevalue}
```

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

```
1422 \newcounter{zpage}
```

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

```
1423 \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`.

```
1424 \let\ZREF@org@stepcounter\stepcounter
```

```
1425 \def\stepcounter#1{%
```

```
1426   \ifcsname @stepcounterhook@#1\endcsname
```

```
1427     \csname @stepcounterhook@#1\endcsname
```

```
1428   \fi
```

```
1429   \ZREF@org@stepcounter{#1}%
```

```
1430 }
```

`\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.

```
1431 \ZREF@IfDefinable\zmakeperpage\def{%
```

```
1432   {%
```

```
1433     \ifnextchar [\ZREF@makeperpage@opt{\ZREF@@makeperpage [\ltx@zero]}%
```

```
1434   }%
```

```
1435 }
```

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

```
1436 \def\ZREF@makeperpage@opt[#1]{%
```

```
1437   \begingroup
```

```
1438     \edef\x{\endgroup
```

```
1439       \noexpand\ZREF@@makeperpage[\number\numexpr#1-1\relax]}%
```

```
1440   }%
```

```
1441   \x
```

```
1442 }
```

```
1443 \def\ZREF@@makeperpage[#1]#2{%
```

```
1444   \ifundefined{@stepcounterhook@#2}{%
```

```
1445     \expandafter\gdef\csname @stepcounterhook@#2\endcsname{}}%
```

```
1446   }{ }%
```

```
1447   \expandafter\gdef\csname ZREF@perpage@#2\endcsname{%
```

```
1448     \ZREF@@perpage@step{#2}{#1}%
```

```
1449   }%
```

```
1450   \expandafter\g@addto@macro\csname @stepcounterhook@#2\endcsname{%
```

```
1451     \ifcsname ZREF@perpage@#2\endcsname
```

```
1452       \csname ZREF@perpage@#2\endcsname
```

```
1453     \fi
```

```
1454   }%
```

```
1455 }
```

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

```
1456 \def\ZREF@@perpage@step#1#2{%
```

First the reference is generated.

```
1457   \global\advance\c@zref@unique\ltx@one
```

```
1458   \begingroup
```

```
1459     \expandafter
```

```
1460     \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.

```

1461   \zref@ifrefundefined\thezref@unique{%
1462     \global\c@zpage=\c@page
1463     \global\let\thezpage\thepage
1464     \expandafter\xdef\csname ZREF@abspage@#1\endcsname{%
1465       \number\c@abspage
1466     }%
1467   }%

```

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

```

1468     \global\c@zpage=\zref@extract\thezref@unique{pagevalue}\relax
1469     \xdef\thezpage{\noexpand\zref@extract{\thezref@unique}{page}}%
1470     \expandafter\xdef\csname ZREF@abspage@#1\endcsname{%
1471       \zref@extractdefault\thezref@unique
1472         {abspage}{\number\c@abspage}%
1473     }%
1474   }%

```

Page changes are detected by a changed absolute page number.

```

1475     \expandafter\ifx\csname ZREF@abspage@#1\endcsname
1476         \csname ZREF@currentabspage@#1\endcsname
1477     \else
1478       \global\csname c@#1\endcsname=#2\relax
1479       \global\expandafter\let
1480         \csname ZREF@currentabspage@#1\endcsname
1481         \csname ZREF@abspage@#1\endcsname
1482     \fi
1483 \endgroup
1484 }

```

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

```

1485 \ZREF@ifDefinable\zunmakeperpage\def{%
1486   #1{%
1487     \global\expandafter
1488     \let\csname ZREF@perpage@#1\endcsname\@undefined
1489   }%
1490 }
1491 </perpage>

```

6.15 Module `titleref`

```

1492 <*titleref>
1493 \NeedsTeXFormat{LaTeX2e}
1494 \ProvidesPackage{zref-titleref}%
1495 [2010/05/01 v2.17 Module titleref for zref (HO)]%
1496 \RequirePackage{zref-base}[2010/05/01]
1497 \ifx\ZREF@base@ok Y%
1498 \else
1499   \expandafter\endinput
1500 \fi
1501 \RequirePackage{getttitlestring}[2009/12/08]

```

6.15.1 Implementation

```
1502 \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`.

`\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`.

```
1503 \let\zref@titleref@current\ltx@empty
```

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

```
1504 \zref@newprop{title}{\zref@titleref@current}%  
1505 \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.

```
1506 \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`.

```
1507 \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.

```
1508 \ZREF@Robust\def\zref@titleref@cleanup#1{%  
1509   \begingroup  
1510   \toks@\expandafter{%  
1511     \ZREF@titleref@hook  
1512     #1%  
1513   }%  
1514   \expandafter\endgroup  
1515   \expandafter\def\expandafter\ZREF@titleref@hook\expandafter{%  
1516     \the\toks@  
1517   }%  
1518 }%
```

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

```
1519 \newif\ifzref@titleref@stripperperiod  
1520 \zref@titleref@stripperperiodtrue
```

`\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.

```
1521 \ZREF@Robust\def\zref@titleref@setcurrent#1{%  
1522   \ifzref@titleref@expand  
1523     \GetTitleStringExpand{#1}%  
1524   \else  
1525     \GetTitleStringNonExpand{#1}%  
1526   \fi  
1527   \edef\zref@titleref@current{%  
1528     \detokenize\expandafter{\GetTitleStringResult}%  
1529   }%  
1530   \ifzref@titleref@stripperperiod  
1531     \edef\zref@titleref@current{%  
1532       \expandafter\ZREF@stripperperiod\zref@titleref@current  
1533       \ltx@empty.\ltx@empty\@nil  
1534     }%  
1535   \fi
```

```

1536 }%
1537 \GetTitleStringDisableCommands{%
1538   \ZREF@titleref@hook
1539 }

```

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

```

1540 \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`.

```

1541 \define@key{ZREF@TR}{expand}[true]{%
1542   \csname zref@titleref@expand#1\endcsname
1543 }%
1544 \define@key{ZREF@TR}{stripperperiod}[true]{%
1545   \csname zref@titleref@stripperperiod#1\endcsname
1546 }%
1547 \define@key{ZREF@TR}{cleanup}{%
1548   \zref@titleref@cleanup{#1}%
1549 }%
1550 \define@key{ZREF@TR}{title}{%
1551   \def\zref@titleref@current{#1}%
1552 }%
1553 \ZREF@IfDefinable\ztitlerefsetup\def{%
1554   {\setkeys{ZREF@TR}}%
1555 }%

```

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

```

1556 \ZREF@IfDefinable\ztitleref\def{%
1557   {\zref@wrapper@babel\ZREF@titleref}%
1558 }%
1559 \def\ZREF@titleref#1{%
1560   \begingroup
1561     \zref@refused{#1}%
1562     \let\label\ltx@gobble
1563     \zref@extract{#1}{title}%
1564   \endgroup
1565 }%

```

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.

```

1566 \AtBeginDocument{%
1567   \ZREF@patch@caption{%
1568     \long\def\@caption#1[#2]{%
1569       \zref@titleref@setcurrent{#2}%
1570       \ZREF@org@caption{#1}[{#2}]%
1571     }%
1572   }%

```

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

```

1573 \ZREF@patch@part{%
1574   \def\@part[#1]{%
1575     \zref@titleref@setcurrent{#1}%
1576     \ZREF@org@part[{#1}]%
1577   }%

```

```

1578 }%
1579 \ZREF@patch{@chapter}{%
1580   \def\@chapter[#1]{%
1581     \zref@titleref@setcurrent{#1}%
1582     \ZREF@org@@chapter[#{1}]%
1583   }%
1584 }%
1585 \ZREF@patch{@sect}{%
1586   \def\@sect#1#2#3#4#5#6[#7]{%
1587     \zref@titleref@setcurrent{#7}%
1588     \ZREF@org@@sect{#1}{#2}{#3}{#4}{#5}{#6}[#{7}]%
1589   }%
1590 }%

```

The star versions of the section commands.

```

1591 \ZREF@patch{@spart}{%
1592   \def\@spart#1{%
1593     \zref@titleref@setcurrent{#1}%
1594     \ZREF@org@@spart{#1}%
1595   }%
1596 }%
1597 \ZREF@patch{@schapter}{%
1598   \def\@schapter#1{%
1599     \zref@titleref@setcurrent{#1}%
1600     \ZREF@org@@schapter{#1}%
1601   }%
1602 }%
1603 \ZREF@patch{@ssect}{%
1604   \def\@ssect#1#2#3#4#5{%
1605     \zref@titleref@setcurrent{#5}%
1606     \ZREF@org@@ssect{#1}{#2}{#3}{#4}{#5}%
1607   }%
1608 }%

```

6.15.4 Environment description

```

1609 \ZREF@patch{descriptionlabel}{%
1610   \def\descriptionlabel#1{%
1611     \zref@titleref@setcurrent{#1}%
1612     \ZREF@org@descriptionlabel{#1}%
1613   }%
1614 }%

```

6.15.5 Class memoir

```

1615 \@ifclassloaded{memoir}{%
1616   \ltx@ifundefined{ifheadnameref}{}{%
1617     \def\@chapter[#1]#2{%
1618       \ltx@ifundefined{ch@pt@c}{%
1619         \zref@titleref@setcurrent{#1}%
1620       }{%
1621         \ifx\ch@pt@c\ltx@empty
1622           \zref@titleref@setcurrent{#2}%
1623         \else
1624           \def\NR@temp{#1}%
1625           \ifx\NR@temp\ltx@empty
1626             \expandafter\zref@titleref@setcurrent
1627             \expandafter{\ch@pt@c}%
1628           \else
1629             \ifheadnameref
1630               \zref@titleref@setcurrent{#1}%
1631             \else
1632               \expandafter\zref@titleref@setcurrent
1633               \expandafter{\ch@pt@c}%

```

```

1634         \fi
1635     \fi
1636     \fi
1637 }%
1638 \ZREF@org@@chapter[#{1}]{#2}%
1639 }%
1640 \ZREF@patch{M@sect}{%
1641     \def\M@sect#1#2#3#4#5#6[#7][#8]{%
1642         \ifheadnameref
1643             \zref@titleref@setcurrent{#8}%
1644         \else
1645             \zref@titleref@setcurrent{#7}%
1646         \fi
1647         \ZREF@org@M@sect{#1}{#2}{#3}{#4}{#5}{#6}[#{7}][#{8}]%
1648     }%
1649 }%
1650 }%
1651 }{}%

```

6.15.6 Class beamer

```

1652 \@ifclassloaded{beamer}{%
1653     \ZREF@patch{beamer@section}{%
1654         \long\def\beamer@section[#1]{%
1655             \zref@titleref@setcurrent{#1}%
1656             \ZREF@org@beamer@section[#{1}]%
1657         }%
1658     }%
1659     \ZREF@patch{beamer@subsection}{%
1660         \long\def\beamer@subsection[#1]{%
1661             \zref@titleref@setcurrent{#1}%
1662             \ZREF@org@beamer@subsection[#{1}]%
1663         }%
1664     }%
1665     \ZREF@patch{beamer@subsubsection}{%
1666         \long\def\beamer@subsubsection[#1]{%
1667             \zref@titleref@setcurrent{#1}%
1668             \ZREF@org@beamer@subsubsection[#{1}]%
1669         }%
1670     }%
1671 }{}%

```

6.15.7 Package titlesec

```

1672 \@ifpackageloaded{titlesec}{%
1673     \ZREF@patch{ttl@sect@i}{%
1674         \def\ttl@sect@i#1#2[#3]#4{%
1675             \zref@titleref@setcurrent{#4}%
1676             \ZREF@org@ttl@sect@i{#1}{#2}[#{3}]{#4}%
1677         }%
1678     }%
1679 }{}%

```

6.15.8 Package longtable

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

```

1680 \@ifpackageloaded{longtable}{%
1681     \ZREF@patch{LT@c@ption}{%
1682         \def\LT@c@ption#1[#2]#3{%
1683             \ZREF@org@LT@c@ption{#1}[#{2}]{#3}%
1684             \zref@titleref@setcurrent{#2}%
1685         }%
1686     }%
1687 }{}%

```

6.15.9 Package listings

Package listings: support for its caption.

```
1688 \@ifpackageloaded{listings}{%
1689   \ZREF@patch{lst@MakeCaption}{%
1690     \def\lst@MakeCaption{%
1691       \ifx\lst@label\ltx@empty
1692       \else
1693         \expandafter\zref@titleref@setcurrent\expandafter{%
1694           \lst@@caption
1695         }%
1696       \fi
1697     \ZREF@org@lst@MakeCaption
1698   }%
1699 }%
1700 }-{}%
```

6.15.10 Theorems

```
1701 \ZREF@patch{@opargbegintheorem}{%
1702   \def\@opargbegintheorem#1#2#3{%
1703     \zref@titleref@setcurrent{#3}%
1704     \ZREF@org@@opargbegintheorem{#1}{#2}{#3}%
1705   }%
1706 }%
1707 \@ifpackageloaded{amsthm}{%
1708   \begingroup
1709   \edef\x{macro:\string#1\string#2[\string#3]}%
1710   \@onelevel@sanitize\x
1711   \def\y#1->#2\@nil{#1}%
1712   \edef\z{\expandafter\y\meaning\@begintheorem->\@nil}%
1713   \@onelevel@sanitize\z
1714   \expandafter\endgroup
1715   \ifx\x\z
1716     \ZREF@patch{@begintheorem}{%
1717       \def\@begintheorem#1#2[#3]{%
1718         \zref@titleref@setcurrent{#3}%
1719         \ZREF@org@@begintheorem{#1}{#2}[{#3}]%
1720       }%
1721     }%
1722   \fi
1723 }-{}%
1724 }
1725 </titleref)
```

6.16 Module xr

```
1726 <*xr>
1727 \NeedsTeXFormat{LaTeX2e}
1728 \ProvidesPackage{zref-xr}%
1729 [2010/05/01 v2.17 Module xr for zref (HO)]%
1730 \RequirePackage{zref-base}[2010/05/01]
1731 \ifx\ZREF@base@ok Y%
1732 \else
1733   \expandafter\endinput
1734 \fi
1735 \RequirePackage{keyval}
1736 \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.

```
1737 \zref@newprop{url}{}%
```

```

1738 \zref@newprop{urluse}{}%
1739 \zref@newprop{externaldocument}{}%

\ZREF@xr@NewPropAnchor
1740 \def\ZREF@xr@NewPropAnchor{%
1741   \ltx@ifpackageloaded{zref-hyper}{%
1742   }{%
1743     \zref@newprop{anchor}{%
1744       \ltx@ifundefined{@currentHref}{}{\@currentHref}%
1745     }%
1746   }%
1747   \global\let\ZREF@xr@NewPropAnchor\relax
1748 }

```

\ZREF@xr@NewPropTitle

```

1749 \def\ZREF@xr@NewPropTitle{%
1750   \ltx@ifpackageloaded{zref-title}{%
1751   }{%
1752     \zref@newprop{title}{}%
1753   }%
1754   \global\let\ZREF@xr@NewPropTitle\relax
1755 }

```

\ZREF@xr@NewPropTheotype

```

1756 \def\ZREF@xr@NewPropTheotype{%
1757   \zref@newprop{theotype}{}%
1758   \global\let\ZREF@xr@NewPropTheotype
1759 }

```

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.

```

1760 \providecommand*\zref@xr@ext{%
1761   \ltx@ifundefined{XR@ext}{pdf}{\XR@ext}%
1762 }%

```

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

```

1763 \newif\ifZREF@xr@zreflabel

1764 \SetupKeyvalOptions{%
1765   family=ZREF@XR,%
1766   prefix=ZREF@xr%
1767 }
1768 \DeclareBoolOption[true]{tozreflabel}
1769 \DeclareBoolOption[false]{toltxlabel}
1770 \DeclareBoolOption{verbose}
1771 \define@key{ZREF@XR}{ext}{%
1772   \def\zref@xr@{#1}%
1773 }
1774 \DeclareBoolOption[false]{urluse}

```

`\zxrsetup`

```

1775 \newcommand*\zxrsetup{%
1776   \setkeys{ZREF@XR}%
1777 }%

```

\ZREF@xr@URL

```

1778 \newcount\ZREF@xr@URL
1779 \ZREF@xr@URL=\ltx@zero

```

`\ZREF@xr@AddURL`

```
1780 \def\ZREF@xr@AddURL#1{%
1781   \begingroup
1782   \def\ZREF@temp{#1}%
1783   \count@=\ltx@one
1784   \ZREF@xr@@AddUrl
1785   \endgroup
1786 }
```

`\ZREF@xr@@AddUrl`

```
1787 \def\ZREF@xr@@AddUrl{%
1788   \ifnum\count@>\ZREF@xr@URL
1789     \global\advance\ZREF@xr@URL by\ltx@one
1790     \xdef\ZREF@xr@theURL{\romannumeral\ZREF@xr@URL}%
1791     \global\expandafter\let
1792       \csname Z@U@\ZREF@xr@theURL\endcsname\ZREF@temp
1793     \@PackageInfo{zref-xr}{%
1794       \ltx@backslashchar Z@U@\ZREF@xr@theURL:\MessageBreak
1795       \ZREF@temp\MessageBreak
1796     }%
1797   \else
1798     \expandafter
1799     \ifx\csname Z@U@\romannumeral\count@\endcsname\ZREF@temp
1800       \xdef\ZREF@xr@theURL{\romannumeral\count@}%
1801     \else
1802       \expandafter\expandafter\expandafter\ZREF@xr@@AddUrl
1803     \fi
1804   \fi
1805 }
```

`\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).

```
1806 \ZREF@IfDefinable\zexternaldocument\def{%
1807   {%
1808     \ZREF@xr@NewPropAnchor
1809     \ZREF@xr@NewPropTitle
1810     \begingroup
1811     \csname @safe@actives@true\endcsname
1812     \makeatletter
1813     \@ifstar{%
1814       \ZREF@xr@zreflabelfalse
1815       \@testopt\ZREF@xr@externaldocument{}%
1816     }{%
1817       \ZREF@xr@zreflabeltrue
1818       \@testopt\ZREF@xr@externaldocument{}%
1819     }%
1820   }%
1821 }
```

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^AT_EX's order using `\input`.

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

```
1822 \def\ZREF@xr@externaldocument[#1]#2{%
1823   \def\ZREF@xr@prefix{#1}%
1824   \let\ZREF@xr@filelist\ltx@empty
1825   \edef\ZREF@xr@externalfile{#2}%
1826   \edef\ZREF@xr@file{\ZREF@xr@externalfile.aux}%

```



```

1827 \filename@parse{#2}%
1828 \@testopt\ZREF@xr@graburl{#2.\zref@xr@ext}%
1829 }%
1830 \def\ZREF@xr@graburl[#1]{%
1831 \edef\ZREF@xr@url{#1}%
1832 \ifZREF@xr@urluse
1833 \expandafter\ZREF@xr@AddURL\expandafter{\ZREF@xr@url}%
1834 \expandafter\def\expandafter\ZREF@xr@url
1835 \expandafter{\csname Z@U@\ZREF@xr@theURL\endcsname}%
1836 \fi
1837 \ZREF@xr@checkfile
1838 \endgroup
1839 }%

```

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

```

1840 \def\ZREF@xr@checkfile{%
1841 \openin\@inputcheck\ZREF@xr@file\relax
1842 \ifeof\@inputcheck
1843 \PackageWarning{zref-xr}{%
1844 File '\ZREF@xr@file' not found or empty,\MessageBreak
1845 labels not imported%
1846 }%
1847 \else
1848 \PackageInfo{zref-xr}{%
1849 Label \ifZREF@xr@zreflabel (zref) \fi
1850 import from '\ZREF@xr@file'%
1851 }%
1852 \def\ZREF@xr@found{0}%
1853 \def\ZREF@xr@ignored@empty{0}%
1854 \def\ZREF@xr@ignored@zref{0}%
1855 \def\ZREF@xr@ignored@ltx{0}%
1856 \ZREF@xr@processfile
1857 \closein\@inputcheck
1858 \begingroup
1859 \let\on@line\ltx@empty
1860 \PackageInfo{zref-xr}{%
1861 Statistics for '\ZREF@xr@file':\MessageBreak
1862 \ZREF@xr@found\space
1863 \ifZREF@xr@zreflabel zref\else LaTeX\fi\space
1864 label(s) found%
1865 \ifnum\ZREF@xr@ignored@empty>0 %
1866 ,\MessageBreak
1867 \ZREF@xr@ignored@empty\space empty label(s) ignored%
1868 \fi
1869 \ifnum\ZREF@xr@ignored@zref>0 %
1870 ,\MessageBreak
1871 \ZREF@xr@ignored@zref\space
1872 duplicated zref label(s) ignored%
1873 \fi
1874 \ifnum\ZREF@xr@ignored@ltx>0 %
1875 ,\MessageBreak
1876 \ZREF@xr@ignored@ltx\space
1877 duplicated latex label(s) ignored%
1878 \fi
1879 }%
1880 \endgroup
1881 \fi
1882 \ifx\ZREF@xr@filelist\ltx@empty
1883 \else
1884 \edef\ZREF@xr@file{%
1885 \expandafter\ltx@car\ZREF@xr@filelist\@nil
1886 }%

```

```

1887 \edef\ZREF@xr@filelist{%
1888 \expandafter\ltx@cdr\ZREF@xr@filelist\ltx@empty\@nil
1889 }%
1890 \expandafter\ZREF@xr@checkfile
1891 \fi
1892 }%

```

\ZREF@xr@processfile

```

1893 \def\ZREF@xr@processfile{%
1894 \read\@inputcheck to\ZREF@xr@line
1895 \expandafter\ZREF@xr@processline\ZREF@xr@line..\ZREF@nil
1896 \ifeof\@inputcheck
1897 \else
1898 \expandafter\ZREF@xr@processfile
1899 \fi
1900 }%

```

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

```

1901 \long\def\ZREF@xr@processline#1#2#3\ZREF@nil{%
1902 \def\x{#1}%
1903 \toks@{#2}%
1904 \ifZREF@xr@zreflabel
1905 \ifx\x\ZREF@xr@zref@newlabel
1906 \expandafter
1907 \ZREF@xr@process@zreflabel\ZREF@xr@line...\ZREF@nil
1908 \fi
1909 \else
1910 \ifx\x\ZREF@xr@newlabel
1911 \expandafter
1912 \ZREF@xr@process@label\ZREF@xr@line... []\ZREF@nil
1913 \fi
1914 \fi
1915 \ifx\x\ZREF@xr@@input
1916 \edef\ZREF@xr@filelist{%
1917 \etex@unexpanded\expandafter{\ZREF@xr@filelist}%
1918 {\filename@area\the\toks@}%
1919 }%
1920 \fi
1921 }%
1922 \def\ZREF@xr@process@zreflabel\zref@newlabel#1#2#3\ZREF@nil{%
1923 \edef\ZREF@xr@refname{Z@R@\ZREF@xr@prefix#1}%
1924 \edef\ZREF@xr@found{\the\numexpr\ZREF@xr@found+1\relax}%
1925 \def\x{#2}%
1926 \edef\ZREF@xr@tempname{temp$}%
1927 \edef\ZREF@xr@temprefname{Z@R@\ZREF@xr@tempname}%
1928 \let\ZREF@xr@list\x
1929 \ifx\ZREF@xr@list\ltx@empty
1930 \PackageWarningNoLine{zref-xr}{%
1931 Label '#1' without properties ignored\MessageBreak
1932 in file '\ZREF@xr@file'%
1933 }%
1934 \edef\ZREF@xr@ignored@empty{%
1935 \the\numexpr\ZREF@xr@ignored@empty+1\relax
1936 }%
1937 \else
1938 \expandafter\ZREF@xr@checklist\x\ZREF@nil
1939 \expandafter\let\csname\ZREF@xr@temprefname\endcsname\x
1940 \expandafter\ltx@LocalAppendToMacro
1941 \csname\ZREF@xr@temprefname\endcsname
1942 \expandafter{%
1943 \expandafter\externaldocument\expandafter{%
1944 \ZREF@xr@externalfile

```



```

2007 }
2008 \def\ZREF@xr@zref@newlabel{\zref@newlabel}%
2009 \def\ZREF@xr@newlabel{\newlabel}%
2010 \def\ZREF@xr@input{\@input}%
2011 \def\ZREF@xr@relax{\relax}%

```

\ZREF@xr@tolabel

```

2012 \def\ZREF@xr@tolabel#1#2{%
2013   \ifZREF@xr@verbose
2014     \PackageInfo{zref-xr}{%
2015       Import to LaTeX label ‘#2’%
2016     }%
2017   \fi
2018   \zref@wrapper@unexpanded{%
2019     \expandafter\xdef\csname r@#2\endcsname{%
2020       {%
2021         \ltx@ifundefined{M@TitleReference}{%
2022           \ltx@ifundefined{TR@TitleReference}{%
2023             \zref@extractdefault{#1}{default}{}}%
2024           }{%
2025             \noexpand\TR@TitleReference
2026             {\zref@extractdefault{#1}{default}{}}%
2027             {\zref@extractdefault{#1}{title}{}}%
2028           }%
2029         }{%
2030           \noexpand\M@TitleReference
2031           {\zref@extractdefault{#1}{default}{}}%
2032           {\zref@extractdefault{#1}{title}{}}%
2033         }%
2034       }%
2035     {\zref@extractdefault{#1}{page}{}}%
2036     \ltx@ifpackageloaded{nameref}{%
2037       {\zref@extractdefault{#1}{title}{}}%
2038       {\zref@extractdefault{#1}{anchor}{}}%
2039       \zref@ifrefcontainsprop{#1}{urluse}{%
2040         {\zref@extractdefault{#1}{urluse}{}}%
2041       }{%
2042         {\zref@extractdefault{#1}{url}{}}%
2043       }%
2044     }{}%
2045   }%
2046 }%
2047 }

```

\ZREF@xr@zref@ignorewarning

```

2048 \def\ZREF@xr@zref@ignorewarning#1{%
2049   \PackageWarningNoLine{zref-xr}{%
2050     Zref label ‘#1’ is already in use\MessageBreak
2051     in file ‘\ZREF@xr@file’%
2052   }%
2053   \edef\ZREF@xr@ignored@empty{%
2054     \the\numexpr\ZREF@xr@ignored+1%
2055   }%
2056 }%

```

\ZREF@xr@ltx@ignorewarning

```

2057 \def\ZREF@xr@ltx@ignorewarning#1{%
2058   \PackageWarningNoLine{zref-xr}{%
2059     LaTeX label ‘#1’ is already in use\MessageBreak
2060     in file ‘\ZREF@xr@file’%
2061   }%
2062   \edef\ZREF@xr@ignored@ltx{%

```

```

2063 \the\numexpr\ZREF@xr@ignored@ltx+1%
2064 }%
2065 }%

```

\ZREF@xr@checklist

```

2066 \def\ZREF@xr@checklist#1#2#3\ZREF@nil{%
2067 \ifx\@undefined#1\relax
2068 \expandafter\ZREF@xr@checkkey\string#1\@nil
2069 \fi
2070 \ifx\#3\%
2071 \else
2072 \ltx@ReturnAfterFi{%
2073 \ZREF@xr@checklist#3\ZREF@nil
2074 }%
2075 \fi
2076 }%
2077 \def\ZREF@xr@checkkey#1#2\@nil{%
2078 \zref@ifpropundefined{#2}{%
2079 \zref@newprop{#2}{}%
2080 }{%
2081 }%

```

\ZREF@xr@scanparams

```

2082 \def\ZREF@xr@scanparams#1#2#3#4#5#6#7\ZREF@nil{%
2083 \let#1\ltx@empty
2084 \ZREF@foundfalse
2085 \ZREF@xr@scantitleref#1#2\TR@TitleReference{}{}%
2086 \ifZREF@found
2087 \else
2088 \ltx@LocalAppendToMacro#1{\default{#2}}%
2089 \fi
2090 % page
2091 \ltx@LocalAppendToMacro#1{\page{#3}}%
2092 % nameref title
2093 \ifZREF@found
2094 \else
2095 \ifx\#4\%
2096 \else
2097 \def\ZREF@xr@temp{#4}%
2098 \ifx\ZREF@xr@temp\ZREF@xr@relax
2099 \else
2100 \ltx@LocalAppendToMacro#1{\title{#4}}%
2101 \fi
2102 \fi
2103 \fi
2104 % anchor
2105 \ifx\#5\%
2106 \else
2107 \ltx@LocalAppendToMacro#1{\anchor{#5}}%
2108 \fi
2109 \ifx\#6\%
2110 \else
2111 \ifZREF@xr@urluse
2112 \ZREF@xr@AddURL{#6}%
2113 \expandafter\ltx@LocalAppendToMacro\expandafter#1%
2114 \expandafter{%
2115 \expandafter\urluse\expandafter{%
2116 \csname Z@U@\ZREF@xr@theURL\endcsname
2117 }%
2118 }%
2119 \else
2120 \ltx@LocalAppendToMacro#1{\url{#6}}%

```

```

2121 \fi
2122 \fi
2123 }%

```

`\ZREF@xr@scantitleref`

```

2124 \def\ZREF@xr@scantitleref#1#2\TR@TitleReference#3#4#5\ZREF@nil{%
2125 \ifx\#5\%
2126 \else
2127 \ltx@LocalAppendToMacro#1{%
2128 \default{#3}%
2129 \title{#4}%
2130 }%
2131 \ZREF@foundtrue
2132 \fi
2133 }%

```

`\ZREF@xr@urlcheck`

```

2134 \def\ZREF@xr@urlcheck#1{%
2135 \zref@ifrefcontainsprop{#1}{anchor}{%
2136 \zref@ifrefcontainsprop{#1}{url}{%
2137 }{%
2138 \expandafter
2139 \ltx@LocalAppendToMacro\csname Z@R@#1\expandafter\endcsname
2140 \expandafter{%
2141 \csname url\ifZREF@xr@urluse use\fi
2142 \expandafter\endcsname\expandafter{\ZREF@xr@url}%
2143 }%
2144 }%
2145 }{%
2146 }%
2147 }%

2148 </xr>

```

6.17 Module `hyperref`

UNFINISHED :-(

```

2149 <*hyperref>
2150 \NeedsTeXFormat{LaTeX2e}
2151 \ProvidesPackage{zref-hyperref}%
2152 [2010/05/01 v2.17 Module hyperref for zref (HO)]%
2153 \RequirePackage{zref-base}[2010/05/01]
2154 \ifx\ZREF@base@ok Y%
2155 \else
2156 \expandafter\endinput
2157 \fi

2158 \ltx@ifpackageloaded{zref-xr}{}{%
2159 \zref@newprop{anchor}[]{}%
2160 \ltx@ifundefined{@currentHref}{}{\@currentHref}%
2161 }%
2162 }
2163 \zref@addprop\ZREF@mainlist{anchor}%
2164 </hyperref>

```

6.18 Module `savepos`

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

6.18.1 Identification

```
2165 <*savepos>
2166 \NeedsTeXFormat{LaTeX2e}
2167 \ProvidesPackage{zref-savepos}%
2168 [2010/05/01 v2.17 Module savepos for zref (HO)]%
2169 \RequirePackage{zref-base}[2010/05/01]
2170 \ifx\ZREF@base@ok Y%
2171 \else
2172 \expandafter\endinput
2173 \fi
```

6.18.2 Availability

First we check, whether the feature is available.

```
2174 \ltx@ifundefined{pdfsavepos}{%
2175 \PackageError\ZREF@name{%
2176 \string\pdfsavepos\space is not supported.\MessageBreak
2177 It is provided by pdfTeX (1.40) or XeTeX%
2178 }\ZREF@UpdatePdfTeX
2179 \endinput
2180 }{}%
```

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 X_{TeX} also provides `\pdfsavepos`.

```
2181 \RequirePackage{ifpdf}
2182 \ifpdf
2183 \else
2184 \ltx@ifundefined{pdftexversion}{%
2185 }{%
2186 \ifnum\pdftexversion<140 %
2187 \PackageError\ZREF@name{%
2188 \string\pdfsavepos\space is not supported in DVI mode%
2189 \MessageBreak
2190 of this pdfTeX version%
2191 }\ZREF@UpdatePdfTeX
2192 \expandafter\expandafter\expandafter\endinput
2193 \fi
2194 }%
2195 \fi
```

6.18.3 Setup

```
2196 \zref@newlist{savepos}
2197 \zref@newprop*{posx}[0]{\the\pdflastxpos}
2198 \zref@newprop*{posy}[0]{\the\pdflastypos}
2199 \zref@addprop{savepos}{posx, posy}
```

6.18.4 User macros

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

```
2200 \ZREF@ifdefinable\zsavapos\def{%
2201 #1{%
2202 \@bsphack
2203 \if@filesw
2204 \pdfsavepos
2205 \zref@labelbylist{#1}{savepos}%
2206 \fi
2207 \@esphack
2208 }%
2209 }
```

`\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 pdf \TeX . Use differences instead. The unit of the position numbers is sp.

```

2210 \newcommand*{\zposx}[1]{%
2211   \zref@extract{#1}{posx}%
2212 }%
2213 \newcommand*{\zposy}[1]{%
2214   \zref@extract{#1}{posy}%
2215 }%

```

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 \TeX about undefined references.

`\ZREF@savepos@ok`

```

2216 \let\ZREF@savepos@ok=Y
2217 </savepos>

```

6.19 Module `abspos`

6.19.1 Identification

```

2218 <*abspos>
2219 \NeedsTeXFormat{LaTeX2e}
2220 \ProvidesPackage{zref-abspos}%
2221   [2010/05/01 v2.17 Module abspos for zref (HO)]%
2222 \RequirePackage{zref-base}[2010/05/01]
2223 \ifx\ZREF@base@ok Y%
2224 \else
2225   \expandafter\endinput
2226 \fi

2227 \RequirePackage{zref-savepos}[2010/05/01]
2228 \ifx\ZREF@savepos@ok Y%
2229 \else
2230   \expandafter\endinput
2231 \fi

2232 \RequirePackage{zref-pagelayout}[2010/05/01]
2233 \zref@addprop{savepos}{abspage}

2234 </abspos>

```

6.20 Module `dotfill`

```

2235 <*dotfill>
2236 \NeedsTeXFormat{LaTeX2e}
2237 \ProvidesPackage{zref-dotfill}%
2238   [2010/05/01 v2.17 Module dotfill for zref (HO)]%
2239 \RequirePackage{zref-base}[2010/05/01]
2240 \ifx\ZREF@base@ok Y%
2241 \else
2242   \expandafter\endinput
2243 \fi

```

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

```

2244 \RequirePackage{zref-savepos}[2010/05/01]

```

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

```

2245 \zref@require@unique

```


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

```
2246 \RequirePackage{keyval}
```

The definitions of the keys follow.

```
2247 \define@key{ZREF@DF}{unit}{%
```

```
2248   \def\ZREF@df@unit{#1}%
```

```
2249 }
```

```
2250 \define@key{ZREF@DF}{min}{%
```

```
2251   \def\ZREF@df@min{#1}%
```

```
2252 }
```

```
2253 \define@key{ZREF@DF}{dot}{%
```

```
2254   \def\ZREF@df@dot{#1}%
```

```
2255 }
```

Defaults are set, see user interface.

```
2256 \providecommand\ZREF@df@min{2}
```

```
2257 \providecommand\ZREF@df@unit{.44em}
```

```
2258 \providecommand\ZREF@df@dot{.}
```

`\zdotfillsetup` Configuration of `\zdotfill` is done by `\zdotfillsetup`.

```
2259 \newcommand*{\zdotfillsetup}{\setkeys{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.

```
2260 \ZREF@IfDefinable\zdotfill\def{%
```

```
2261   {%
```

```
2262     \leavevmode
```

```
2263     \global\advance\c@zref@unique\ltx@one
```

```
2264     \begingroup
```

```
2265     \def\ZREF@temp{zref@\number\c@zref@unique}%
```

```
2266     \pdfsavepos
```

```
2267     \zref@labelbyprops{\thezref@unique L}{posx}%
```

```
2268     \setlength{\dimen@}{\ZREF@df@unit}%
```

```
2269     \zref@ifrefundefined{\thezref@unique R}{%
```

```
2270       \ZREF@dotfill
```

```
2271     }{%
```

```
2272       \ifnum\numexpr\zposx{\thezref@unique R}%
```

```
2273         -\zposx{\thezref@unique L}\relax
```

```
2274         <\dimexpr\ZREF@df@min\dimen@\relax
```

```
2275         \hfill
```

```
2276       \else
```

```
2277         \ZREF@dotfill
```

```
2278       \fi
```

```
2279     }{%
```

```
2280     \pdfsavepos
```

```
2281     \zref@labelbyprops{\thezref@unique R}{posx}%
```

```
2282   \endgroup
```

```
2283   \kern\z@
```

```
2284 }{%
```

```
2285 }
```

`\ZREF@dotfill` Help macro that actually sets the dots.

```
2286 \def\ZREF@dotfill{%
```

```
2287   \cleaders\hb@xt@\dimen@{\hss\ZREF@df@dot\hss}\hfill
```

```
2288 }
```

```
2289 </dotfill>
```

7 Test

7.1 `\zref@localaddprop`

```
2290 <*test1>
```

```

2291 \NeedsTeXFormat{LaTeX2e}
2292 \nofiles
2293 \documentclass{article}
2294 \usepackage{zref-base}[2010/05/01]
2295 \usepackage{qstest}
2296 \IncludeTests{*}
2297 \LogTests{log}{*}{*}
2298
2299 \makeatletter
2300 \def\ExpectList#1#2{%
2301   \expandafter\expandafter\expandafter\Expect
2302   \expandafter\expandafter\expandafter{\csname Z@L@#1\endcsname}{#2}%
2303 }
2304 \begin{qstest}{localaddprop}{localaddprop}
2305   \ExpectList{main}{\default\page}%
2306   \Expect{undefined}*{\meaning\foobar}%
2307   \zref@newprop{foobar}{F00}%
2308   \Expect{undefined}*{\meaning\foobar}%
2309   \zref@newlist{alist}%
2310   \ExpectList{alist}{}%
2311   \begin{group}
2312     \zref@localaddprop{main}{foobar}%
2313     \Expect{undefined}*{\meaning\foobar}%
2314     \ExpectList{main}{\default\page\foobar}%
2315     \zref@localaddprop{alist}{page}%
2316     \ExpectList{alist}{\page}%
2317   \end{group}
2318   \ExpectList{main}{\default\page}%
2319   \ExpectList{alist}{}%
2320   \zref@addprop{alist}{foobar}%
2321   \ExpectList{alist}{\foobar}%
2322   \Expect{undefined}*{\meaning\foobar}%
2323 \end{qstest}
2324 \@@end
2325 </test1>

```

7.2 Module base

```

2326 <*test-base>
2327 \NeedsTeXFormat{LaTeX2e}
2328 \documentclass{article}
2329 \usepackage{zref-base,zref-titleref}[2010/05/01]
2330 \usepackage{qstest}
2331 \IncludeTests{*}
2332 \LogTests{log}{*}{*}
2333
2334 \makeatletter
2335 \newcommand*\DefExpand[2]{%
2336   \expandafter\expandafter\expandafter\def
2337   \expandafter\expandafter\expandafter#1%
2338   \expandafter\expandafter\expandafter{#2}%
2339   \@onelevel@sanitize#1%
2340 }
2341 \newcommand*\Test[3]{%
2342   \Expect{#2}*{#1}%
2343   \zref@wrapper@unexpanded{%
2344     \Expect*{#3}*{#1}%
2345   }%
2346   \DefExpand\x{#1}%
2347   \Expect*{#3}*{\x}%
2348 }
2349 \makeatother
2350

```

```

2351 \begin{document}
2352 \section{\textit{Hello} \textbf{World}}
2353 \label{sec:hello}
2354 \makeatletter
2355 \zref@newprop{foo}[\@empty D\@empty efault]{\@empty V\@empty alue}
2356 \begin{qstest}{getcurrent}{getcurrent}
2357 \Test{\zref@getcurrent{foo}}%
2358     {Value}{\noexpand\@empty V\noexpand\@empty alue}%
2359 \Test{\zref@getcurrent{xy}}{-}{-}%
2360 \end{qstest}
2361 \begin{qstest}{extract}{extract}
2362 \def\textbf#1{<#1>}%
2363 \def\textit#1{[#1]}% hash-ok
2364 \Test{\zref@extractdefault{xy}{page}{\@empty D\@empty efault}}%
2365     {Default}{\noexpand\@empty D\noexpand\@empty efault}%
2366 \Test{\zref@extractdefault{sec:hello}{foo}{\@empty A\@empty B}}%
2367     {AB}{\noexpand\@empty A\noexpand\@empty B}%
2368 \Test{\zref@extract{sec:hello}{foo}}%
2369     {Default}{\noexpand\@empty D\noexpand\@empty efault}%
2370 \zref@ifrefundefined{sec:hello}{%
2371 }{%
2372 \Test{\zref@extract{sec:hello}{default}}{1}{1}%
2373 \Test{\zref@extract{sec:hello}{title}}%
2374     {[Hello] <World>}%
2375     {\noexpand\textit{Hello} \noexpand\textbf{World}}%
2376 }%
2377 \end{qstest}
2378 \end{document}
2379 </test-base>

```

7.3 Module runs

```

2380 <*test-runs>
2381 \NeedsTeXFormat{LaTeX2e}
2382 \documentclass{article}
2383 \usepackage{zref-runs}[2010/05/01]
2384 \usepackage{qstest}
2385 \IncludeTests{*}
2386 \LogTests{log}{*}{*}
2387
2388 \begin{qstest}{zruns-preamble}{zruns-preamble}
2389 \Expect{0}*{\zruns}%
2390 \end{qstest}
2391
2392 \AtBeginDocument{%
2393 \begin{qstest}{zruns-atbegindocument}{zruns-atbegindocument}%
2394 \Expect*{\number\ExpectRuns}*{\zruns}%
2395 \end{qstest}%
2396 }
2397
2398 \begin{document}
2399 \begin{qstest}{zruns-document}{zruns-document}
2400 \Expect*{\number\ExpectRuns}*{\zruns}%
2401 \end{qstest}
2402 \end{document}
2403 </test-runs>

```

7.4 Module titleref

```

2404 <*test-titleref-memoir>
2405 \NeedsTeXFormat{LaTeX2e}
2406 \documentclass{memoir}
2407 \usepackage{zref-titleref}[2010/05/01]
2408 \usepackage{qstest}

```

```

2409 \IncludeTests{*}
2410 \LogTests[log]{*}{*}
2411 \begin{document}
2412 \makeatletter
2413 \def\List{}
2414 \def\Label#1{%
2415   \zref@label{#1}%
2416   \g@addto@macro\List{%
2417     \par
2418     #1: [\ztitleref{#1}]%
2419   }%
2420   \mbox{}%
2421   \zref@refused{#1}%
2422   \zref@ifrefundefined{#1}{%
2423   }{%
2424     \begingroup
2425       \edef\x{\zref@extract{#1}{title}}%
2426       \Expect{OK/}*{\expandafter\ltx@carthree\x{}{}{}\@nil}%
2427     \endgroup
2428   }%
2429 }
2430 \def\Test#1{%
2431   \csname#1\endcsname*{OK/#1}%
2432   \Label{#1*}%
2433   \csname#1\endcsname{OK/#1}%
2434   \Label{#1}%
2435   \csname#1\endcsname[OK/#1-toc]%
2436           {WRONG-in-titleref/#1-toc-2}%
2437   \Label{#1-toc}%
2438   \expandafter\ifx\csname#1\endcsname\part
2439   \else
2440     \headnamereffalse
2441     \csname#1\endcsname[OK/#1-th-toc]%
2442           [WRONG-in-titleref/#1-th-toc-2]%
2443           {WRONG-in-titleref/#1-th-toc-3}%
2444     \Label{#1-th-toc}%
2445     \headnamereftrue
2446     \csname#1\endcsname[WRONG-in-titleref/#1-th-head-1]%
2447           [OK/#1-th-head]%
2448           {WRONG-in-titleref/#1-th-head-3}%
2449     \Label{#1-th-head}%
2450   \fi
2451 }
2452 \begin{qstest}{section}{section}
2453   \@for\x:=part,chapter,section,subsection,subsubsection\do{%
2454     \expandafter\Test\expandafter{x}%
2455   }%
2456 \end{qstest}
2457 \newpage
2458 \List
2459 \end{document}
2460 </test-titleref-memoir>

```

8 Installation

8.1 Download

Package. This package is available on CTAN²:

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

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

²[ftp://ftp.ctan.org/tex-archive/](http://ftp.ctan.org/tex-archive/)

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):

<code>zref.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref.sty</code>
<code>zref-base.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-base.sty</code>
<code>zref-abspage.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-abspage.sty</code>
<code>zref-abspos.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-abspos.sty</code>
<code>zref-counter.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-counter.sty</code>
<code>zref-dotfill.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-dotfill.sty</code>
<code>zref-hyperref.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-hyperref.sty</code>
<code>zref-lastpage.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-lastpage.sty</code>
<code>zref-marks.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-marks.sty</code>
<code>zref-nextpage.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-nextpage.sty</code>
<code>zref-pageattr.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-pageattr.sty</code>
<code>zref-pagelayout.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-pagelayout.sty</code>
<code>zref-perpage.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-perpage.sty</code>
<code>zref-runs.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-runs.sty</code>
<code>zref-savepos.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-savepos.sty</code>
<code>zref-thepage.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-thepage.sty</code>
<code>zref-titleref.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-titleref.sty</code>
<code>zref-totpages.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-totpages.sty</code>
<code>zref-user.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-user.sty</code>
<code>zref-xr.sty</code>	\rightarrow <code>tex/latex/oberdiek/zref-xr.sty</code>
<code>zref.pdf</code>	\rightarrow <code>doc/latex/oberdiek/zref.pdf</code>
<code>zref-example.tex</code>	\rightarrow <code>doc/latex/oberdiek/zref-example.tex</code>
<code>zref-example-lastpage.tex</code>	\rightarrow <code>doc/latex/oberdiek/zref-example-lastpage.tex</code>
<code>zref-example-nextpage.tex</code>	\rightarrow <code>doc/latex/oberdiek/zref-example-nextpage.tex</code>
<code>test/zref-test1.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/zref-test1.tex</code>
<code>test/zref-test-base.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/zref-test-base.tex</code>
<code>test/zref-test-runs.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/zref-test-runs.tex</code>
<code>test/zref-test-titleref-memoir.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/zref-test-titleref-memoir.tex</code>
<code>zref.dtx</code>	\rightarrow <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 ($\text{t}_{\text{e}}\text{T}_{\text{E}}\text{X}$, $\text{m}_{\text{i}}\text{k}_{\text{T}}\text{E}_{\text{X}}$, ...) relies on file name databases, you must refresh these. For example, $\text{t}_{\text{e}}\text{T}_{\text{E}}\text{X}$ users run `texhash` or `mktextlsr`.

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 $\text{X}_{\text{T}}\text{E}\text{X}$.

[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 ε -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 ‘external-document’.

[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 `toltxlabel` 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@tolabel`.
- `\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.

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	
<code>\@@end</code>	2324
<code>\@PackageError</code>	400, 416
<code>\@PackageInfo</code>	1793
<code>\@PackageInfoNoLine</code>	998, 1074, 1160, 1173, 1185, 1241, 1286
<code>\@addtoreset</code>	680, 752
<code>\@auxout</code>	506
<code>\@begintheorem</code>	1712, 1717
<code>\@bsphack</code>	462, 472, 2202
<code>\@caption</code>	1568
<code>\@chapter</code>	1580, 1617
<code>\@currentHref</code>	1744, 2160
<code>\@currentlabel</code>	704
<code>\@ehc</code>	294, 304, 383, 406, 418, 1317
<code>\@empty</code>	2355, 2358, 2364, 2365, 2366, 2367, 2369
<code>\@esphack</code>	469, 489, 2207
<code>\@for</code>	2453
<code>\@ifclassloaded</code>	1615, 1652
<code>\@ifdefinable</code>	240, 287
<code>\@ifnextchar</code>	422, 1433
<code>\@ifpackageloaded</code>	1672, 1680, 1688, 1707
<code>\@ifstar</code>	387, 1813
<code>\@ifundefined</code>	192, 678, 1444, 1949, 1991
<code>\@input</code>	2010
<code>\@inputcheck</code>	1841, 1842, 1857, 1894, 1896
<code>\@latex@warning</code>	568
<code>\@mainaux</code>	1402
<code>\@namedef</code>	427
<code>\@newl@bel</code>	283
<code>\@nil</code> .	1332, 1334, 1533, 1540, 1711, 1712, 1885, 1888, 2068, 2077, 2426
<code>\@onelevel@sanitize</code>	397, 425, 1710, 1713, 2339
<code>\@opargbegintheorem</code>	1702
<code>\@part</code>	1574
<code>\@schapter</code>	1598
<code>\@sect</code>	1586
<code>\@spart</code>	1592
<code>\@ssect</code>	1604
<code>\@testdef</code>	1056, 1057, 1220, 1221
<code>\@testopt</code>	1815, 1818, 1828
<code>\@tfor</code>	320, 511
<code>\@undefined</code>	1488, 2067
<code>\@</code>	25, 26, 27, 28, 153, 155, 157, 158, 170, 173, 1976, 2070, 2095, 2105, 2109, 2125
<code>_</code>	44, 45
A	
<code>\AddLineBeginAux</code>	278
<code>\advance</code>	797, 1113, 1268, 1457, 1789, 2263
<code>\afterassignment</code>	231, 877, 881
<code>\AfterLastShipout</code>	794, 1130, 1293
<code>\Alph</code>	7
<code>\anchor</code>	2107
<code>\AtBeginDocument</code>	773, 966, 1396, 1566, 2392
<code>\AtBeginShipout</code>	756, 838
<code>\AtEndOfPackage</code>	195
<code>\AtVeryEndDocument</code>	1065, 1229
B	
<code>\beamer@section</code>	1654
<code>\beamer@subsection</code>	1660
<code>\beamer@subsubsection</code>	1666
<code>\begin</code>	23, 57, 100, 106, 156, 172, 2304, 2351, 2356, 2361, 2388, 2393, 2398, 2399, 2411, 2452
<code>\bfseries</code>	697
C	
<code>\c@abspage</code>	759, 1465, 1472
<code>\c@page</code>	797, 1419, 1462
<code>\c@zpage</code>	1462, 1468
<code>\c@zref@unique</code> .	684, 1457, 2263, 2265
<code>\ch@pt@c</code>	1621, 1627, 1633
<code>\chapter</code>	24, 30, 32, 61, 82
<code>\ChapterPages</code>	91, 112
<code>\ChapterStart</code>	78, 135, 150, 166
<code>\ChapterStop</code>	85, 148, 165, 184
<code>\chardef</code>	899, 914, 923, 927
<code>\cleaders</code>	2287
<code>\cleardoublepage</code>	79, 86
<code>\clearpage</code>	62
<code>\closein</code>	1857
<code>\columnsep</code>	1047
<code>\columnwidth</code>	1046

<code>\comma@entry</code> . . .	341, 342, 344, 350, 360, 361, 363, 369, 476, 478, 482, 1338, 1339, 1340, 1346, 1349	2019, 2116, 2139, 2142, 2302, 2431, 2433, 2435, 2438, 2441, 2446
<code>\comma@parse</code>	340, 359, 475, 1337	
<code>\count@</code>	1071, 1082, 1083, 1085, 1112, 1113, 1122, 1124, 1125, 1238, 1259, 1261, 1262, 1267, 1268, 1783, 1788, 1799, 1800	<code>\endinput</code> 192, 262, 275, 716, 748, 769, 791, 832, 868, 959, 977, 1141, 1197, 1304, 1416, 1499, 1733, 2156, 2172, 2179, 2192, 2225, 2230, 2242
<code>\csname</code>	250, 251, 288, 313, 314, 315, 324, 349, 350, 368, 369, 428, 430, 431, 436, 440, 449, 466, 482, 521, 529, 535, 578, 579, 581, 601, 624, 625, 626, 665, 1061, 1225, 1367, 1375, 1427, 1445, 1447, 1450, 1452, 1464, 1470, 1475, 1476, 1478, 1480, 1481, 1488, 1542, 1545, 1792, 1799, 1811, 1835, 1939, 1941, 1956, 1957, 1974, 1981, 1984, 1998, 1999, 2019, 2116, 2139, 2141, 2302, 2431, 2433, 2435, 2438, 2441, 2446	<code>\escapechar</code> . . 327, 513, 986, 1013, 1148 <code>\etex@unexpanded</code> . 453, 610, 630, 1917 <code>\evensidemargin</code> 1038 <code>\Expect</code> 2301, 2306, 2308, 2313, 2322, 2342, 2344, 2347, 2389, 2394, 2400, 2426 <code>\ExpectList</code> 2300, 2305, 2310, 2314, 2316, 2318, 2319, 2321 <code>\ExpectRuns</code> 2394, 2400 <code>\externaldocument</code> 1943, 1985
<code>\current@chapid</code>	80, 88	
D		
<code>\DeclareBoolOption</code>	1768, 1769, 1770, 1774	
<code>\DeclareOption</code>	194	
<code>\default</code>	2088, 2128, 2305, 2314, 2318	
<code>\DefExpand</code>	2335, 2346	
<code>\define@key</code>	1541, 1544, 1547, 1550, 1771, 2247, 2250, 2253	
<code>\descriptionlabel</code>	1610	
<code>\detokenize</code>	1528	
<code>\dftest</code>	167, 174, 175, 176, 177, 178, 179, 180, 181, 182	
<code>\dimen@</code>	2268, 2274, 2287	
<code>\dimexpr</code>	153, 155, 1125, 2274	
<code>\directlua</code>	990, 1152	
<code>\do</code>	325, 511, 2453	
<code>\documentclass</code>	4, 39, 68, 270, 2293, 2328, 2382, 2406	
<code>\dotfill</code>	169, 173	
E		
<code>\emph</code>	150	
<code>\end</code>	34, 64, 130, 159, 183, 185, 2323, 2360, 2377, 2378, 2390, 2395, 2401, 2402, 2456, 2459	
<code>\endcsname</code>	250, 251, 288, 313, 314, 315, 324, 349, 350, 368, 369, 428, 430, 431, 436, 440, 449, 466, 482, 527, 529, 535, 578, 579, 581, 601, 624, 625, 626, 652, 665, 1061, 1225, 1367, 1375, 1426, 1427, 1445, 1447, 1450, 1451, 1452, 1464, 1470, 1475, 1476, 1478, 1480, 1481, 1488, 1542, 1545, 1792, 1799, 1811, 1835, 1939, 1941, 1956, 1957, 1974, 1981, 1984, 1998, 1999,	
F		
<code>\fancyhead</code>	51, 54	
<code>\fancyhf</code>	50, 53	
<code>\fancypagestyle</code>	52	
<code>\filename@area</code>	1918	
<code>\filename@parse</code>	1827	
<code>\foo</code>	18, 29, 31, 33	
<code>\foobar</code>	2306, 2308, 2313, 2314, 2321, 2322	
<code>\footskip</code>	1043	
<code>\frontmatter</code>	58, 103	
G		
<code>\g@addto@macro</code>	348, 1450, 2416	
<code>\G@refundefinedtrue</code>	567	
<code>\gdef</code>	431, 436, 1403, 1445, 1447	
<code>\GetTitleStringDisableCommands</code>	1537	
<code>\GetTitleStringExpand</code>	1523	
<code>\GetTitleStringNonExpand</code>	1525	
<code>\GetTitleStringResult</code>	1528	
H		
<code>\hb@xt@</code>	2287	
<code>\headheight</code>	1041	
<code>\headnamereffalse</code>	2440	
<code>\headnamereftrue</code>	2445	
<code>\headsep</code>	1042	
<code>\hfill</code>	2275, 2287	
<code>\hoffset</code>	1034	
<code>\hss</code>	2287	
I		
<code>\if@filesw</code>	501, 795, 1401, 2203	
<code>\if@safe@actives</code>	658	
<code>\ifcase</code>	115, 935, 1338	
<code>\ifcsname</code>	652, 1426, 1451	
<code>\ifeof</code>	1842, 1896	
<code>\ifetex@unexpanded</code>	265	
<code>\ifheadnameref</code>	1629, 1642	
<code>\ifin@</code>	315	
<code>\ifluatex</code>	982, 1144	
<code>\ifnum</code>	809, 910, 920, 926, 983, 1112, 1145, 1267, 1310, 1363, 1788, 1865, 1869, 1874, 2186, 2272	
<code>\ifodd</code>	124	
<code>\ifpdf</code>	2182	

<code>\ifx</code>	399, 520, 586, 714, 719, 746, 767, 789, 830, 866, 957, 975, 1060, 1139, 1196, 1205, 1224, 1302, 1320, 1329, 1333, 1338, 1339, 1340, 1414, 1475, 1497, 1621, 1625, 1691, 1715, 1731, 1799, 1882, 1905, 1910, 1915, 1929, 1976, 2067, 2070, 2095, 2098, 2105, 2109, 2125, 2154, 2170, 2223, 2228, 2240, 2438	1150, 1172, 1184, 1203, 1204, 1232, 1275, 1616, 1618, 2174, 2184	
<code>\ifZREF@found</code>	245, 2086, 2093	<code>\ltx@ifundefined</code>	298, 377, 445, 551, 596, 995, 1014, 1055, 1157, 1219, 1744, 1761, 2021, 2022, 2160
<code>\ifZREF@immediate</code>	491, 503, 507, 522	<code>\ltx@LocalAppendToMacro</code>	367, 1285, 1940, 1980, 1983, 2088, 2091, 2100, 2107, 2113, 2120, 2127, 2139
<code>\ifZREF@pa@list</code>	1212, 1217	<code>\ltx@newif</code>	1048, 1212
<code>\ifZREF@pl@list</code>	1048, 1053	<code>\ltx@one</code>	1113, 1268, 1457, 1783, 1789, 2263
<code>\ifzref@titleref@expand</code>	1506, 1522	<code>\ltx@ReturnAfterFi</code>	2072
<code>\ifzref@titleref@stripperiod</code>	1519, 1530	<code>\ltx@secondoftwo</code>	309, 575, 588, 618, 655, 659, 813
<code>\ifZREF@xr@toltxlabel</code>	1962, 2004	<code>\ltx@space</code>	446, 448, 597, 606, 620, 623, 920, 926, 1084, 1123, 1312, 1340, 1383
<code>\ifZREF@xr@tozreflabel</code>	1948, 1990	<code>\ltx@undefined</code>	989, 1151
<code>\ifZREF@xr@urluse</code>	1832, 2111, 2141	<code>\ltx@zero</code>	1310, 1363, 1433, 1779
<code>\ifZREF@xr@verbose</code>	1950, 1992, 2013	<code>\luatexversion</code>	983, 1145
<code>\ifZREF@xr@zreflabel</code>	1763, 1849, 1863, 1904		
<code>\immediate</code>	496, 1402		
<code>\in@</code>	312		
<code>\IncludeTests</code>	2296, 2331, 2385, 2409		
<code>\item</code>	107, 111, 113, 121, 125, 127		
	K		
<code>\kern</code>	2283		
<code>\kv@key</code>	1332, 1334, 1335, 1349		
<code>\kv@parse</code>	1328		
<code>\kv@value</code>	1329, 1330, 1337		
	L		
<code>\Label</code>	2414, 2432, 2434, 2437, 2444, 2449		
<code>\label</code>	719, 1562, 2353		
<code>\leavevmode</code>	2262		
<code>\List</code>	2413, 2416, 2458		
<code>\LogTests</code>	2297, 2332, 2386, 2410		
<code>\lst@caption</code>	1694		
<code>\lst@label</code>	1691		
<code>\lst@MakeCaption</code>	1690		
<code>\LT@c@ption</code>	1682		
<code>\ltx@backslashchar</code>	534, 1237, 1287, 1794		
<code>\ltx@car</code>	1332, 1885		
<code>\ltx@carthree</code>	2426		
<code>\ltx@c@dr</code>	1334, 1888		
<code>\ltx@empty</code>	288, 391, 510, 587, 1320, 1503, 1507, 1533, 1540, 1621, 1625, 1691, 1824, 1859, 1882, 1888, 1929, 2083		
<code>\ltx@firstofone</code>	252, 636, 647, 653, 1206, 1207		
<code>\ltx@firstoftwo</code>	590, 617, 618, 661, 811		
<code>\ltx@gobble</code>	248, 353, 372, 485, 719, 720, 1115, 1270, 1351, 1562		
<code>\ltx@gobbletwo</code>	680, 752, 1353		
<code>\ltx@ifpackageloaded</code>	1741, 1750, 2036, 2158		
<code>\ltx@ifUndefined</code>	227, 247, 255, 646, 688, 843, 988,		
		M	
		<code>\m@ne</code>	797
		<code>\M@s@ct</code>	1641
		<code>\M@TitleReference</code>	2030
		<code>\mag</code>	1025
		<code>\mainmatter</code>	60, 134
		<code>\makeatletter</code>	11, 74, 101, 1812, 2299, 2334, 2354, 2412
		<code>\makeatother</code>	16, 99, 2349
		<code>\makebox</code>	169, 170
		<code>\MakeRobustcommand</code>	230
		<code>\marginparsep</code>	1045
		<code>\marginparwidth</code>	1044
		<code>\mbox</code>	2420
		<code>\meaning</code>	1712, 2306, 2308, 2313, 2322
		<code>\MessageBreak</code>	268, 405, 1070, 1082, 1086, 1126, 1237, 1260, 1263, 1287, 1289, 1313, 1316, 1343, 1345, 1384, 1385, 1794, 1795, 1844, 1861, 1866, 1870, 1875, 1931, 2050, 2059, 2176, 2189
		N	
		<code>\NeedsTeXFormat</code>	3, 188, 220, 710, 742, 763, 783, 826, 862, 953, 971, 1135, 1298, 1392, 1410, 1493, 1727, 2150, 2166, 2219, 2236, 2291, 2327, 2381, 2405
		<code>\newcommand</code>	18, 78, 85, 91, 167, 718, 725, 876, 893, 894, 963, 1307, 1775, 2210, 2213, 2259, 2335, 2341
		<code>\newcount</code>	1778
		<code>\newcounter</code>	6, 681, 753, 1422
		<code>\newif</code>	245, 491, 1506, 1519, 1763
		<code>\newlabel</code>	1967, 1978, 2009
		<code>\newmarks</code>	1316
		<code>\newpage</code>	143, 2457
		<code>\nfss@text</code>	697
		<code>\nofiles</code>	2292
		<code>\NR@temp</code>	1624, 1625

<code>\number</code>	94, 109, 684, 689, 844, 1018, 1019, 1397, 1419, 1439, 1465, 1472, 2265, 2394, 2400	<code>\reset@font</code>	697
<code>\numexpr</code>	94, 109, 115, 691, 846, 907, 1397, 1439, 1924, 1935, 1969, 2054, 2063, 2272	<code>\rightarrow</code>	45
		<code>\romannumeral</code>	444, 595, 616, 1321, 1790, 1799, 1800
O		S	
<code>\oddsidemargin</code>	1037	<code>\section</code>	63, 137, 145, 2352
<code>\on@line</code>	1399, 1859	<code>\setcounter</code>	755
<code>\openin</code>	1841	<code>\setkeys</code>	1554, 1776, 2259
P		<code>\setlength</code>	2268
<code>\PackageError</code>	256, 267, 292, 302, 381, 1311, 2175, 2187	<code>\SetupKeyvalOptions</code>	1764
<code>\PackageInfo</code>	289, 412, 1400, 1848, 1860, 1951, 1993, 2014	<code>\space</code>	569, 1174, 1186, 1862, 1863, 1867, 1871, 1876, 2176, 2188
<code>\PackageWarning</code>	343, 362, 477, 1341, 1382, 1843	<code>\stepcounter</code>	19, 757, 1424, 1425
<code>\PackageWarningNoLine</code>	1930, 2049, 2058	<code>\stockheight</code>	1029
<code>\page</code>	2091, 2305, 2314, 2316, 2318	<code>\stockwidth</code>	1028
<code>\pagestyle</code>	49	T	
<code>\paperheight</code>	1027	<code>\tableofcontents</code>	59, 132
<code>\paperwidth</code>	1026	<code>\Test</code>	2341, 2357, 2359, 2364, 2366, 2368, 2372, 2373, 2430, 2454
<code>\par</code>	2417	<code>\textbf</code>	2352, 2362, 2375
<code>\part</code>	2438	<code>\textheight</code>	1040
<code>\pdf@escapehex</code>	1209	<code>\textit</code>	2352, 2363, 2375
<code>\pdf@unescapehex</code>	1210	<code>\textwidth</code>	1039
<code>\pdfhorigin</code>	1006, 1032	<code>\the</code>	13, 153, 155, 482, 488, 533, 547, 691, 759, 803, 840, 846, 907, 1054, 1074, 1081, 1082, 1083, 1085, 1122, 1124, 1125, 1180, 1192, 1218, 1242, 1258, 1259, 1261, 1516, 1918, 1924, 1935, 1969, 2054, 2063, 2197, 2198
<code>\pdflastxpos</code>	2197	<code>\thechapter</code>	14
<code>\pdflastypos</code>	2198	<code>\thefoo</code>	7, 12, 20
<code>\pdfpageattr</code>	1166, 1174, 1180	<code>\thetype</code>	1981
<code>\pdfpageheight</code>	1005, 1031	<code>\thepage</code>	43, 44, 45, 504, 508, 569, 705, 1463
<code>\pdfpagesattr</code>	1167, 1186, 1192	<code>\thezpage</code>	16, 1463, 1469
<code>\pdfpagewidth</code>	1004, 1030	<code>\thezref@unique</code>	10, 683, 1460, 1461, 1468, 1469, 1471, 2267, 2269, 2272, 2273, 2281
<code>\pdfsavepos</code>	2176, 2188, 2204, 2266, 2280	<code>\title</code>	2100, 2129
<code>\pdftexversion</code>	2186	<code>\toks@</code>	474, 481, 482, 488, 531, 533, 546, 547, 798, 803, 1070, 1074, 1080, 1081, 1237, 1242, 1257, 1258, 1510, 1516, 1903, 1918
<code>\pdfvorigin</code>	1007, 1033	<code>\topmargin</code>	1036
<code>\ProcessOptions</code>	217	<code>\TR@TitleReference</code>	2025, 2085, 2124
<code>\protect</code>	567	<code>\ttl@sect@i</code>	1674
<code>\protected</code>	236	U	
<code>\protected@write</code>	506	<code>\unexpanded</code>	268, 273
<code>\providecommand</code>	279, 1395, 1760, 2256, 2257, 2258	<code>\UniqueCounterCall</code>	891
<code>\ProvidesPackage</code>	189, 221, 711, 743, 764, 784, 827, 863, 954, 972, 1136, 1299, 1393, 1411, 1494, 1728, 2151, 2167, 2220, 2237	<code>\UniqueCounterNew</code>	874
R		<code>\url</code>	2120
<code>\read</code>	1894	<code>\urluse</code>	2115
<code>\refstepcounter</code>	775	<code>\usepackage</code>	9, 41, 48, 70, 72, 2294, 2295, 2329, 2330, 2383, 2384, 2407, 2408
<code>\renewcommand</code>	7, 46, 683		
<code>\RequirePackage</code>	191, 196, 223, 224, 225, 228, 264, 269, 277, 713, 745, 750, 766, 786, 787, 788, 829, 834, 835, 865, 870, 871, 872, 873, 956, 961, 962, 974, 979, 980, 981, 1138, 1143, 1179, 1191, 1199, 1200, 1201, 1301, 1306, 1413, 1418, 1496, 1501, 1502, 1730,		

V

\value 13, 840, 1054, 1218
 \verb 173
 \voffset 1035

W

\write 495, 496, 1402

X

\x 328, 333, 514, 519, 666, 668, 933,
 949, 1015, 1022, 1073, 1076,
 1079, 1111, 1240, 1245, 1256,
 1266, 1365, 1370, 1373, 1379,
 1438, 1441, 1709, 1710, 1715,
 1902, 1905, 1910, 1915, 1925,
 1928, 1938, 1939, 1970, 1975,
 2346, 2347, 2425, 2426, 2453, 2454
 \XR@ext 1761

Y

\y 1711, 1712

Z

\z 1712, 1713, 1715
 \z@ 2283
 \Z@D@page 894
 \Z@L@LastPage 800
 \Z@L@main 799
 \zdotfill 17, 170, 173, 2260
 \zdotfillsetup 18, 2259
 \zexternaldocument 18, 1806
 \ziflastpage 12, 816
 \zifrefundefined 7, 553
 \zlabel 11, 83, 104, 138, 146, 718
 \zlistpageattr 1213
 \zlistpagelayout 15, 1049
 \zmakeperpage 16, 1431
 \znextpage 13, 51, 54, 890
 \znextpagesetup 14, 42, 876
 \znonextpagename 46, 893, 941
 \zpageref 11, 126, 734
 \zposx 17, 153, 2210, 2272, 2273
 \zposy 17, 155, 2210
 \zref 11, 25, 26, 27, 28, 112,
 114, 123, 128, 129, 139, 725, 735
 \ZREF@@@newprop 430, 434
 \ZREF@@@extract 599, 605
 \ZREF@@@makeperpage .. 1433, 1439, 1443
 \ZREF@@@newprop 408, 419, 422, 426
 \ZREF@@@perpage@step 1448, 1456
 \zref@addprop
 . 5, 15, 76, 338, 706, 760, 772,
 837, 1020, 1181, 1193, 1349,
 1421, 1505, 2163, 2199, 2233, 2320
 \ZREF@addtoks 545
 \ZREF@base@ok
 . 707, 714, 746, 767, 789, 830,
 866, 957, 975, 1139, 1302, 1414,
 1497, 1731, 2154, 2170, 2223, 2240
 \ZREF@call ... 899, 914, 923, 927, 935
 \zref@default ... 8, 422, 597, 694, 696
 \ZREF@df@dot 2254, 2258, 2287
 \ZREF@df@min 2251, 2256, 2274
 \ZREF@df@unit 2248, 2257, 2268
 \ZREF@dotfill 2270, 2277, 2286
 \ZREF@extract 594, 611, 614, 644
 \zref@extract 7, 95,
 96, 109, 140, 594, 614, 639, 644,
 732, 849, 945, 1085, 1124, 1125,
 1249, 1279, 1468, 1469, 1563,
 2211, 2214, 2368, 2372, 2373, 2425
 \ZREF@extractdefault 615, 631, 634, 643
 \zref@extractdefault 7, 116,
 117, 607, 634, 638, 643, 809,
 810, 903, 918, 964, 1471, 2023,
 2026, 2027, 2031, 2032, 2035,
 2037, 2038, 2040, 2042, 2364, 2366
 \ZREF@foundfalse 2084
 \ZREF@foundtrue 2131
 \ZREF@getcurrent .. 443, 454, 457, 642
 \zref@getcurrent
 6, 457, 637, 642, 2357, 2359
 \zref@hex 1180, 1192, 1206, 1209
 \ZREF@IfDefinable 239, 553, 734, 737,
 816, 854, 890, 1049, 1213, 1431,
 1485, 1553, 1556, 1806, 2200, 2260
 \ZREF@iflastpage 817, 819, 819
 \zref@iflastpage 12, 808, 822
 \zref@iflistcontainsprop
 6, 307, 342, 361
 \zref@iflistundefined
 5, 286, 297, 301, 308
 \zref@ifpropundefined ... 6, 376,
 380, 410, 476, 618, 1120, 1362, 2078
 \ZREF@ifrefcontainsprop ... 577, 585
 \zref@ifrefcontainsprop
 ... 7, 573, 1122, 2039, 2135, 2136
 \ZREF@ifrefundefined
 555, 558, 900, 911, 921
 \zref@ifrefundefined 7,
 550, 560, 566, 574, 617, 912,
 1083, 1261, 1461, 2269, 2370, 2422
 \ZREF@immediatetrue 494
 \ZREF@label 464, 488, 500, 803
 \zref@label 6, 458, 722, 2415
 \zref@labelbylist
 6, 459, 461, 840, 1460, 2205
 \zref@labelbyprops
 7, 88, 471, 898, 2267, 2281
 \zref@listexists
 5, 300, 319, 339, 358, 463
 \zref@listforloop 318
 \zref@listpageattr 1213
 \zref@listpagelayout 1049
 \zref@localaddprop . 5, 357, 2312, 2315
 \ZREF@mainlist 459,
 700, 703, 706, 760, 772, 1505, 2163
 \ZREF@makeperpage@opt ... 1433, 1436
 \ZREF@MARKS@DefineProp
 1325, 1326, 1327, 1361
 \zref@marks@register
 1307, 1312, 1344, 1383
 \ZREF@name 226, 256, 267,
 289, 292, 302, 343, 362, 381,
 400, 412, 416, 477, 1311, 2175, 2187

<code>\ZREF@NAME@bot</code>	1340, 1360	<code>\ZREF@pagenum@this</code>	902, 907, 910, 920, 926
<code>\ZREF@NAME@first</code>	1339, 1359	<code>\ZREF@par</code>	399, <u>424</u>
<code>\ZREF@NAME@top</code>	1338, 1358	<code>\ZREF@patch</code>	<u>246</u> , 774, 1567, 1573, 1579, 1585, 1591, 1597, 1603, 1609, 1640, 1653, 1659, 1665, 1673, 1681, 1689, 1701, 1716
<code>\zref@newlabel</code>	7, 279, <u>282</u> , 540, 1922, 2008	<code>\zref@pdfpageattr</code>	1176, 1262
<code>\zref@newlist</code>	5, <u>285</u> , 703, 793, 836, 1335, 1420, 2196, 2309	<code>\zref@pdfpageattr@used</code>	1177
<code>\ZREF@newprop</code>	389, 392, <u>395</u>	<code>\zref@pdfpagesattr</code>	1188, <u>1277</u> , 1288
<code>\zref@newprop</code>	6, 12, 13, 14, 75, <u>386</u> , 704, 705, 759, 771, 1017, 1180, 1192, 1366, 1374, 1419, 1504, 1737, 1738, 1739, 1743, 1752, 1757, 2079, 2159, 2197, 2198, 2307, 2355	<code>\zref@pdfpagesattr@used</code>	1189, <u>1282</u>
<code>\ZREF@nextpage</code>	891, 895	<code>\ZREF@pl@AfterLastShipout</code>	<u>1052</u> , <u>1131</u>
<code>\ZREF@nil</code>	436, 587, 626, 1895, 1901, 1907, 1912, 1922, 1938, 1967, 1975, 2066, 2073, 2082, 2085, 2124	<code>\ZREF@pl@AtVeryEnd</code>	1065, <u>1068</u>
<code>\ZREF@NOVALUE</code>	593	<code>\ZREF@pl@ListEntry</code>	1087, 1088, 1089, 1090, 1091, 1092, 1093, 1094, 1095, 1096, 1097, 1098, 1099, 1100, 1101, 1102, 1103, 1104, 1105, 1106, 1107, 1108, <u>1119</u>
<code>\ZREF@novalue</code>	586, 587, 593	<code>\ZREF@pl@ListPage</code>	1072, <u>1078</u>
<code>\ZREF@np@call@next</code>	885, 889, 944	<code>\ZREF@pl@listtrue</code>	1050
<code>\ZREF@np@call@nonext</code>	882, 888, 940	<code>\zref@prop</code>	321, 329, 330, 334
<code>\ZREF@np@call@unknown</code>	878, 887, 936	<code>\zref@propexists</code>	6, 341, 360, <u>379</u> , 439, 726
<code>\ZREF@np@setup@i</code>	877, 880	<code>\ZREF@refname@next</code>	905, 912, 921, 945
<code>\ZREF@np@setup@ii</code>	881, 884	<code>\ZREF@refname@this</code>	897, 898, 900, 903
<code>\ZREF@number</code>	<u>688</u> , 1249, 1253, 1309	<code>\ZREF@RefPrefix</code>	<u>281</u> , 283, 1060, 1224
<code>\ZREF@org@@begintheorem</code>	1719	<code>\ZREF@refused</code>	563, <u>565</u>
<code>\ZREF@org@@caption</code>	1570	<code>\zref@refused</code>	7, 559, <u>562</u> , 731, 738, 820, 821, 852, 967, 1253, 1283, 1561, 2421
<code>\ZREF@org@@chapter</code>	1582, 1638	<code>\zref@require@unique</code>	10, <u>677</u> , 1423, 2245
<code>\ZREF@org@@opargbegintheorem</code>	1704	<code>\ZREF@Robust</code>	<u>229</u> , <u>235</u> , 241, 282, 285, 300, 307, 338, 357, 379, 386, 438, 458, 461, 471, 492, 562, 635, 651, 677, 693, 699, 851, 1252, 1282, 1508, 1521
<code>\ZREF@org@@part</code>	1576	<code>\ZREF@savepos@ok</code>	<u>2216</u> , 2228
<code>\ZREF@org@@schapter</code>	1600	<code>\zref@setcurrent</code>	6, 81, 432, <u>438</u> , 776
<code>\ZREF@org@@sect</code>	1588	<code>\zref@setdefault</code>	8, <u>693</u> , 696
<code>\ZREF@org@@spart</code>	1594	<code>\zref@setmainlist</code>	8, <u>699</u>
<code>\ZREF@org@@ssect</code>	1606	<code>\ZREF@STAR</code>	1333, 1357
<code>\ZREF@org@beamer@section</code>	1656	<code>\ZREF@stripperperiod</code>	1532, <u>1540</u>
<code>\ZREF@org@beamer@subsection</code>	1662	<code>\ZREF@temp</code>	193, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 230, 231, 510, 531, 532, 540, 987, 1004, 1005, 1006, 1007, 1011, 1025, 1026, 1027, 1028, 1029, 1030, 1031, 1032, 1033, 1034, 1035, 1036, 1037, 1038, 1039, 1040, 1041, 1042, 1043, 1044, 1045, 1046, 1047, 1059, 1060, 1149, 1166, 1167, 1171, 1182, 1194, 1196, 1202, 1203, 1204, 1205, 1223, 1224, 1332, 1333, 1782, 1792, 1795, 1799, 2265
<code>\ZREF@org@beamer@subsubsection</code>	1668	<code>\ZREF@TempName</code>	1308, 1320, 1321, 1323, 1349, 1362, 1366, 1374, 1385
<code>\ZREF@org@descriptionlabel</code>	1612	<code>\ZREF@TempNum</code>	1309, 1310, 1314, 1321, 1363, 1376
<code>\ZREF@org@lst@MakeCaption</code>	1697		
<code>\ZREF@org@LT@c@Option</code>	1683		
<code>\ZREF@org@M@sect</code>	1647		
<code>\ZREF@org@refstepcounter</code>	777		
<code>\ZREF@org@stepcounter</code>	1424, 1429		
<code>\ZREF@org@testdef</code>	1056, 1058, 1220, 1222		
<code>\ZREF@org@thepage</code>	504, 508		
<code>\ZREF@org@ttl@sect@i</code>	1676		
<code>\ZREF@org@write</code>	495, 496		
<code>\ZREF@P</code>	396, 397, 399, 401, 410, 413, 417, 427, 428, 430, 431, 432, 436, 511, 515, 516, 525, 529, 534, 535		
<code>\ZREF@pa@AfterLastShipout</code>	<u>1216</u> , 1294		
<code>\ZREF@pa@AtVeryEnd</code>	1229, <u>1232</u> , 1285		
<code>\ZREF@pa@ListPage</code>	1239, <u>1255</u>		
<code>\ZREF@pa@listtrue</code>	1214		
<code>\ZREF@page@max</code>	1054, 1112, 1218, 1267		
<code>\zref@pageattr</code>	<u>1247</u>		
<code>\zref@pageattr@used</code>	<u>1252</u>		
<code>\ZREF@pagenum@last</code>	917, 920		

<code>\zref@thepage</code>	13, 848 , 857	<code>\ZREF@xr@ignored@empty</code>	1853, 1865, 1867, 1934, 1935, 2053
<code>\zref@thepage@name</code>	13, 843 , 849 , 852 , 906	<code>\ZREF@xr@ignored@ltx</code>	1855, 1874, 1876, 2062, 2063
<code>\zref@thepage@refused</code>	851 , 856	<code>\ZREF@xr@ignored@zref</code>	1854, 1869, 1871
<code>\ZREF@titleref</code>	1557, 1559	<code>\ZREF@xr@line</code>	1894, 1895, 1907, 1912
<code>\zref@titleref@cleanup</code>	1508 , 1548	<code>\ZREF@xr@list</code>	1928, 1929
<code>\zref@titleref@current</code>	1503 , 1504 , 1527 , 1531 , 1532 , 1551	<code>\ZREF@xr@ltx@ignorewarning</code>	2057
<code>\ZREF@titleref@hook</code>	1507 , 1511 , 1515 , 1538	<code>\ZREF@xr@newlabel</code>	1910, 2009
<code>\zref@titleref@setcurrent</code>	1521 , 1569 , 1575 , 1581 , 1587 , 1593 , 1599 , 1605 , 1611 , 1619 , 1622 , 1626 , 1630 , 1632 , 1643 , 1645 , 1655 , 1661 , 1667 , 1675 , 1684 , 1693 , 1703 , 1718	<code>\ZREF@xr@NewPropAnchor</code>	1740 , 1808
<code>\zref@titleref@stripperiodtrue</code>	1520	<code>\ZREF@xr@NewPropTheotype</code>	1756 , 1979
<code>\ZREF@u@getcurrent</code>	452	<code>\ZREF@xr@NewPropTitle</code>	1749 , 1809
<code>\zref@unhex</code>	1207, 1210, 1248, 1278	<code>\ZREF@xr@prefix</code>	1823, 1923, 1959, 1963, 1968, 1994, 2001, 2005
<code>\ZREF@UpdatePdfTeX</code>	244, 2178, 2191	<code>\ZREF@xr@process@label</code>	1912, 1967
<code>\ZREF@wrapper@babel</code>	668, 674	<code>\ZREF@xr@process@zreflabel</code>	1907, 1922
<code>\zref@wrapper@babel</code>	9, 140, 555 , 563 , 651 , 722 , 727 , 817 , 1557	<code>\ZREF@xr@processfile</code>	1840 , 1893
<code>\zref@wrapper@immediate</code>	10, 87, 492 , 802, 839	<code>\ZREF@xr@processline</code>	1895 , 1901
<code>\ZREF@wrapper@unexpanded</code>	635 , 649	<code>\ZREF@xr@refname</code>	1923, 1949, 1956, 1968, 1991, 1998
<code>\zref@wrapper@unexpanded</code>	10, 636 , 641 , 646 , 2018, 2343	<code>\ZREF@xr@relax</code>	2011, 2098
<code>\ZREF@wu@extract</code>	609 , 639	<code>\ZREF@xr@scanparams</code>	1973, 2082
<code>\ZREF@wu@extractdefault</code>	629 , 638	<code>\ZREF@xr@scantitleref</code>	2085 , 2124
<code>\ZREF@wu@getcurrent</code>	452 , 637	<code>\ZREF@xr@temp</code>	2097, 2098
<code>\ZREF@X</code>	388 , 391 , 428	<code>\ZREF@xr@tempname</code>	1926, 1927, 1947, 1952, 1963, 1971, 1972, 1989, 2005
<code>\zref@xr@</code>	1772	<code>\ZREF@xr@temprefname</code>	1927, 1939, 1941, 1957, 1972, 1974, 1981, 1984, 1999
<code>\ZREF@xr@@AddUrl</code>	1784 , 1787	<code>\ZREF@xr@theURL</code>	1790, 1792, 1794, 1800, 1835, 2116
<code>\ZREF@xr@@input</code>	1915, 2010	<code>\ZREF@xr@tolabel</code>	1963, 2005, 2012
<code>\ZREF@xr@AddURL</code>	1780 , 1833 , 2112	<code>\ZREF@xr@URL</code>	1778 , 1788 , 1789 , 1790
<code>\ZREF@xr@checkfile</code>	1837 , 1840 , 1890	<code>\ZREF@xr@url</code>	1831, 1833, 1834, 2142
<code>\ZREF@xr@checkkey</code>	2068, 2077	<code>\ZREF@xr@urlcheck</code>	1947, 1989, 2134
<code>\ZREF@xr@checklist</code>	1938, 2066	<code>\ZREF@xr@zref@ignorewarning</code>	1959, 2001, 2048
<code>\zref@xr@ext</code>	19, 1760 , 1828	<code>\ZREF@xr@zref@newlabel</code>	1905, 2008
<code>\ZREF@xr@externaldocument</code>	1815 , 1818 , 1822	<code>\ZREF@xr@zref@labelfalse</code>	1814
<code>\ZREF@xr@externalfile</code>	1825 , 1826 , 1944 , 1986	<code>\ZREF@xr@zref@labeltrue</code>	1817
<code>\ZREF@xr@file</code>	1826 , 1841 , 1844 , 1850 , 1861 , 1884 , 1932 , 2051 , 2060	<code>\ZREF@zref</code>	727 , 730
<code>\ZREF@xr@filelist</code>	1824 , 1882 , 1885 , 1887 , 1888 , 1916 , 1917	<code>\zrefused</code>	11, 92, 93, 161, 162, 163, 737
<code>\ZREF@xr@found</code>	1852 , 1862 , 1924 , 1969	<code>\zruns</code>	15, 1395 , 2389 , 2394 , 2400
<code>\ZREF@xr@graburl</code>	1828 , 1830	<code>\zsavepos</code>	17, 157, 158, 2200
<code>\ZREF@xr@ignored</code>	2054	<code>\zthepage</code>	13, 854
		<code>\ztitleref</code>	16, 1556 , 2418
		<code>\ztitlerefsetup</code>	17, 1541
		<code>\ztotpages</code>	14, 124, 963
		<code>\zunknownnextpagename</code>	13, 894, 937
		<code>\zunmakeperpage</code>	16, 1485
		<code>\zxrsetup</code>	18, 1775