

zhnumber 宏包

李清

sobenlee@gmail.com

2016/05/14 v2.4*

第 1 节 简介

zhnumber 宏包用于将阿拉伯数字按照中文格式输出。相比于 CJKnumb, 它提供的四个格式转换命令 \zhnumber, \zhdigits, \zhnum 和 \zhdig 都是可以适当展开的, 可以正常使用于 PDF 书签和交叉引用。

zhnumber 支持 GBK, Big5 和 UTF8 编码, 依赖 L^AT_EX3 项目的 expl3, xparse 和 l3keys2e 宏包。

第 2 节 使用方法

encoding

Updated: 2014-09-09

encoding = $\langle \text{GBK} | \text{Big5} | \text{UTF8} \rangle$

用于指定编码的宏包选项, 可以在调用宏包的时候设定, 也可以用 \zhnumsetup 在导言区内设定。对于 upL^AT_EX、X_YL^AT_EX 和 LuaL^AT_EX, 不用指定编码, 宏包将自动使用 UTF8 编码。只有 L^AT_EX 和 pdfL^AT_EX 需要指定编码, 如果没有指定, 默认将使用 GBK。

\zhnumber ☆

Updated: 2014-09-09

\zhnumber $\{\langle number \rangle\}$

以中文格式输出数字。这里的数字可以是整数、小数和分数。例如

二十亿零一千二百零二万零一百二十
二十亿零一千二百零二万零一百二十
二十亿零一千二百零二万零一百二十
二千零一十二点零二零一二零
二千零一十二点零
零点二零一二
二万零一百二十分之二万零一百二十
二千零一十二分之零
零分之二千零一十二
二百零一又一百二十分之二千零二十

```
1 \zhnumber{2012020120}\\\n2 \zhnumber{2 012 020 120}\\\n3 \zhnumber{2,012,020,120}\\\n4 \zhnumber{2012.020120}\\\n5 \zhnumber{2012.}\\\n6 \zhnumber{.2012}\\\n7 \zhnumber{20120/20120}\\\n8 \zhnumber{/2012}\\\n9 \zhnumber{2012/}\\\n10 \zhnumber{201;2020/120}
```

\zhdigits ☆

Updated: 2014-09-09

\zhdigits $\{\langle number \rangle\}$
\zhdigits * $\{\langle number \rangle\}$

将阿拉伯数字转换为中文数字串。缺省状态下, \zhdigits 将 0 映射为〇, 如果需要将其映射为零, 可以使用带星号的形式。例如

二〇一二〇二〇一二〇
二零一二零二零一二零

```
1 \zhdigits{2012020120}\\\n2 \zhdigits*{2012020120}
```

*ctex-kit rev. e19bfb4.

<hr/> <code>\zhnum</code> ☆ <hr/>	<code>\zhnum {⟨counter⟩}</code> <code>\pagenumbering {zhnum}</code> Updated: 2016-05-01	与 <code>\roman</code> 等类似,用于将 L ^A T _E X 计数器的值转换为中文数字。例如	
	二	1	<code>\zhnum{section}</code>
<hr/> <code>\zhdig</code> ☆ <hr/>	<code>\zhdig {⟨counter⟩}</code> <code>\pagenumbering {zhdig}</code> New: 2016-05-01	与 <code>\roman</code> 等类似,用于将 L ^A T _E X 计数器的值转换为中文数字串。例如	
	二	1	<code>\zhdig{section}</code>
<hr/> <code>\zhweekday</code> ☆ <hr/>	<code>\zhweekday {⟨yyyy/mm/dd⟩}</code> New: 2012-05-25	输出日期当天的星期。例如	
	星期日	1	<code>\zhweekday{2012/5/20}</code>
<hr/> <code>\zhdate</code> ☆ <hr/>	<code>\zhdate {⟨yyyy/mm/dd⟩}</code> <code>\zhdate * {⟨yyyy/mm/dd⟩}</code> New: 2012-05-25	以中文格式输出日期,其中带 * 的命令还输出星期。例如	
	2012 年 5 月 21 日	1	<code>\zhdate{2012/5/21}\</code>
	2012 年 5 月 21 日星期一	2	<code>\zhdate*{2012/5/21}</code>
<hr/> <code>\zhtoday</code> ☆ <hr/>	与 <code>\today</code> 类似,以中文输出当天的日期。例如 New: 2012-05-25	2016 年 5 月 14 日	1 <code>\zhtoday</code>
<hr/> <code>\zhtime</code> ☆ <hr/>	<code>\zhtime {⟨hh:mm⟩}</code> New: 2012-05-25	以中文格式输出时间。例如	
	23 时 56 分	1	<code>\zhtime{23:56}</code>
<hr/> <code>\zhcurrtime</code> ☆ <hr/>	输出当前的时间。例如 New: 2012-05-25	13 时 38 分	1 <code>\zhcurrtime</code>
<hr/> <code>\zhtiangan</code> ☆ <hr/>	<code>\zhtiangan {⟨number⟩}</code> New: 2015-05-20	输出对应的天干计数。⟨number⟩ 的正常范围是 1–10,超出范围的数字将输出空值。例如	
	甲 乙 丙 丁 戊 癸	1	<code>\zhtiangan{1} \zhtiangan{2} \zhtiangan{3}</code>
		2	<code>\zhtiangan{4} \zhtiangan{5} \zhtiangan{10}</code>
<hr/> <code>\zhdizhi</code> ☆ <hr/>	<code>\zhdizhi {⟨number⟩}</code> New: 2015-05-20	输出对应的地支计数。⟨number⟩ 的正常范围是 1–12,超出范围的数字将输出空值。例如	
	子 丑 寅 卯 辰 亥	1	<code>\zhdizhi{1} \zhdizhi{2} \zhdizhi{3}</code>
		2	<code>\zhdizhi{4} \zhdizhi{5} \zhdizhi{12}</code>
<hr/> <code>\zhgan zhi</code> ☆ <hr/>	<code>\zhgan zhi {⟨number⟩}</code> New: 2015-05-20	输出对应的干支计数。⟨number⟩ 的正常范围是 1–60,超出范围的数字将输出空值。例如	
	甲子 乙丑 丙寅	1	<code>\zhgan zhi{1} \zhgan zhi{2} \zhgan zhi{3} \</code>
	丁卯 戊辰 癸亥	2	<code>\zhgan zhi{4} \zhgan zhi{5} \zhgan zhi{60}</code>

<code>\zhganzhinian</code> ☆	<code>\zhganzhinian {<year>}</code>
New: 2015-05-20	输出公元纪年 <code><year></code> 对应的干支纪年。公元前的年份用负数表示。例如 戊戌 乙卯 甲子 丙申
	<pre> 1 \zhganzhinian{1898} \zhganzhinian{-246} \\ 2 \zhganzhinian{-2697} \zhganzhinian{\year} </pre>
<code>\zhnumExtendScaleMap</code>	<code>\zhnumExtendScaleMap [<character>] {<character₁>, <character₂>, ..., <character_n>}</code>
New: 2012-05-25	缺省状态下 <code>\zhnumber</code> 能正确中文格式化的最大整数是 $10^{48} - 1$, <code>\zhdigits</code> 不受这个大小的限制。可以通过 <code>\zhnumExtendScaleMap</code> 来扩展 <code>\zhnumber</code> 。 <code><character_i></code> 设置 $10^{4(i+11)}$ 。若给出可选项 <code><character></code> , 则当数字大于 $10^{4(n+12)} - 1$ 时, 统一用 <code><character></code> 设置输出数字的进位。
<code>\zhnumsetup</code>	<code>\zhnumsetup {<key₁>=<val₁>, <key₂>=<val₂>, ...}</code>
	用于在导言区或文档中, 设置中文数字的输出格式。目前可以设置的 <code><key></code> 如下介绍。以粗体表示选项的默认值。
<code>time</code>	<code>time = <Arabic Chinese></code>
New: 2012-05-25	设置日期和时间的数字格式, <code><Arabic></code> 为阿拉伯数字, 而 <code><Chinese></code> 为中文数字。例如 二〇一六年五月十四日十三时三十八分
	<pre> 1 \zhnumsetup{time=Chinese} 2 \zhtoday\zhcurrtime </pre>
<code>arabicsep</code>	<code>arabicsep = {<sep>}</code>
New: 2016-05-01	设置日期和时间的数字格式为阿拉伯数字时, 阿拉伯数字与汉字的间隔内容。默认为一个空格。
<code>style</code>	<code>style = <Simplified Traditional Normal Financial Ancient></code>
Updated: 2012-05-25	意义分别为 Simplified 以简体中文输出数字(对 Big5 编码无效); Traditional 以繁体中文输出数字(对 Big5 编码无效); Normal 以小写形式输出中文数字; Financial 以大写形式输出中文数字; Ancient 以廿输出 20, 以卅输出 30, 以卌输出 40, 以佰输出 200。
	可以设置 <code>style</code> 为其中一个, 也可以是前三个与后两个的适当组合, 默认是简体小写。例如 陸萬貳仟零壹拾貳點叁 廿一
	<pre> 1 \zhnumsetup{style={Traditional,Financial}} 2 \zhnumber{62012.3}\\ 3 \zhnumsetup{style=Ancient} 4 \zhnumber{21} </pre>
<code>null</code>	<code>null = <true false></code>
	缺省状态下, 除了 <code>\zhdigits</code> 外, 其它的格式转换命令, 将 0 映射成零, 如果需要将 0 映射成 O, 可以使用这个选项。

ganzhi-cyclicNew: 2015-05-20

ganzhi-cyclic = `<true|false>`

天干、地支和干支的数字都有一定范围。若参数大于这个范围，`\tiangan` 等将输出空值。可以将本选项设置为 `true`，对超出范围的数字取相应的模。请注意，数字 0 的结果总是为空值。例如

甲 乙 壬 癸 壬 辛
子 亥 戌 亥 戌 酉
甲子 乙亥 辛酉
癸亥 壬戌 乙卯

```
1 \zhnumsetup{ganzhi-cyclic}
2 \zhtiangan{11} \zhtiangan{12} \zhtiangan{209}
3 \zhtiangan{-1} \zhtiangan{-2} \zhtiangan{-683} \\
4 \zhdizhi{13} \zhdizhi{24} \zhdizhi{1211}
5 \zhdizhi{-1} \zhdizhi{-2} \zhdizhi{-8199} \\
6 \zhganzhi{61} \zhganzhi{72} \zhganzhi{2158} \\
7 \zhganzhi{-1} \zhganzhi{-2} \zhganzhi{-789}
```

zhnumber 提供下列选项来控制阿拉伯数字的中文映射。

```
- -0 0 1 2 3 4 5 6 7 8 9 10 20 30 40 100 200 1000
E2 E3 E4 E8 E12 E16 E20 E24 E28 E32 E36 E40 E44
F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F100 F1000 FE2 FE3
T1 T2 T3 T4 T5 T6 T7 T8 T9 T10
D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12
GZ1 GZ2 GZ3 GZ4 GZ5 GZ6 GZ7 GZ8 GZ9 GZ10 ... GZ60
dot and parts
year month day hour minute weekday mon tue wed thu fri sat sun
```

其中 `-` 设置负，`-0` 设置 〇，`dot` 设置小数的点，`and` 和 `parts` 分别设置分数的“又”和“分之”，`En` 设置 10^n ，`Fn` 设置数字 n 的大写，`Tn` 设置数字 n 的天干，`Dn` 设置数字 n 的地支，而 `GZn` 设置数字 n 的干支。其它的选项同字面意思，不再赘述。例如

```
\zhnumsetup{2={两}}
```

可以将 2 映射成两。需要说明的是，**zhnumber** 将优先使用这里的设置，所以可能会影响到 `style` 选项。如果要恢复 `style` 的功能，可以使用 `reset` 选项。

resetUpdated: 2014-09-12

reset

用于恢复 **zhnumber** 对阿拉伯数字的初始化映射。**zhnumber** 的中文数字初始化设置见源代码(第 4 节)。

activecharNew: 2014-09-09

activechar = `<true|false>`

在 \LaTeX 或者 \pdf\LaTeX 下面输出汉字，传统的办法需要将汉字的首字节设置为活动字符，然后再通过特殊的宏技巧来实现。因此，**zhnumber** 在载入配置文件的时候，默认会将汉字的首字节设置为活动字符。禁用本选项将不会改变汉字首字节的类代码。需要在本选项之后，使用 `encoding` 或者 `reset` 选项才会有效果。

\zhnumber**\zhdigits****\zhnum****\zhdig**

Updated: 2016-05-01

```
\zhnumber    [{options}] {<number>}
\zhdigits *   [{options}] {<number>}
\zhnum        [{options}] {<counter>}
\zhdig        [{options}] {<counter>}
```

如果只改变当前数字的中文输出格式，可以使用带选项的格式转换命令，其中 `<options>` 与 `\zhnumsetup` 的参数相同，如上所介绍。这些带了选项的命令是不可展开的，在某些场合使用时要小心。

第 3 节 *zhnumber* 宏包代码实现

```
1 <*package>
2 <@=@=zhnum>
3 \msg_new:nnn { zhnumber } { 13-too-old }
4 {
```

```

5      Support~package~'expl3'~too~old. \\\
6      Please~update~an~up~to~date~version~of~the~bundles\\\
7      'l3kernel'~and~'l3packages'\\\
8      using~your~TeX~package~manager~or~from~CTAN.
9  }
10 \ifpackagelater { expl3 } { 2015/09/24 } { }
11 { \msg_error:nn { zhnumber } { l3-too-old } }
12 \RequirePackage { xparse , l3keys2e }

```

\zhnumber 用于将输入的数字按照中文格式输出。

```

13 \DeclareExpandableDocumentCommand \zhnumber { +o +m }
14 {
15   \IfNoValueTF {#1}
16   { \zhnum_number:f }
17   { \zhnumberwithoptions {#1} }
18   {#2}
19 }

```

(End definition for \zhnumber. This function is documented on page 4.)

\zhnumberwithoptions 带选项的用户函数。

```

20 \NewDocumentCommand \zhnumberwithoptions { +m +m }
21 {
22   \group_begin:
23   \keys_set:nn { zhnum / options } {#1}
24   \zhnum_number:f {#2}
25   \group_end:
26 }

```

(End definition for \zhnumberwithoptions.)

\zhnum_number:n 先判断输入的是小数还是分数。

```

\__zhnum_number:www 27 \cs_new:Npn \zhnum_number:n #1
28 { \__zhnum_number:www #1 . \q_nil . \q_stop }
29 \cs_new:Npn \__zhnum_number:www #1 . #2 . #3 \q_stop
30 {
31   \quark_if_nil:nTF {#2}
32   { \__zhnum_integer_or_fraction:www #1 / \q_nil / \q_stop }
33   { \zhnum_decimal:nn {#1} {#2} }
34 }
35 \cs_generate_variant:Nn \zhnum_number:n { f }

```

(End definition for \zhnum_number:n.)

__zhnum_integer_or_fraction:www 判断是否输入的是分数。

```

36 \cs_new:Npn \__zhnum_integer_or_fraction:www #1 / #2 / #3 \q_stop
37 {
38   \quark_if_nil:nTF {#2}
39   { \zhnum_integer:n {#1} }
40   { \__zhnum_fraction:www #2 \q_mark #1 ; \q_nil ; \q_stop }
41 }

```

(End definition for __zhnum_integer_or_fraction:www.)

__zhnum_fraction:www 对分数进行预处理。

```

42 \cs_new:Npn \__zhnum_fraction:www #1 \q_mark #2 ; #3 ; #4 \q_stop
43 {
44   \quark_if_nil:nTF {#3}
45   {
46     \zhnum_blank_to_zero:n {#1}
47     \c_zhnum_parts_tl
48     \zhnum_blank_to_zero:n {#2}
49   }
50   {
51     \tl_if_blank:nF {#2}
52     {

```

```

53         \zhnum_number:n {#2}
54         \c__zhnum_and_tl
55     }
56     \zhnum_blank_to_zero:n {#1}
57     \c__zhnum_parts_tl
58     \zhnum_blank_to_zero:n {#3}
59 }
60 }

```

(End definition for `__zhnum_fraction:www`.)

`\zhnum_decimal:nn` 对小数进行预处理。

```

61 \cs_new:Npn \zhnum_decimal:nn #1#2
62 {
63     \zhnum_blank_to_zero:n {#1} \c__zhnum_dot_tl
64     \tl_if_blank:nTF {#2}
65     { \c__zhnum_zero_tl }
66     { \zhnum_digits_zero:n {#2} }
67 }

```

(End definition for `\zhnum_decimal:nn`.)

`\zhnum_blank_to_zero:n` 输出小数的整数位。

```

68 \cs_new:Npn \zhnum_blank_to_zero:n #1
69 {
70     \tl_if_blank:nTF {#1}
71     { \c__zhnum_zero_tl }
72     { \zhnum_number:n {#1} }
73 }

```

(End definition for `\zhnum_blank_to_zero:n`.)

`\zhnum` 用于将 L^AT_EX 计数器按中文格式输出。

```

\zhnumberwithoptions
74 \DeclareExpandableDocumentCommand \zhnum { +o +m }
75 {
76     \IfNoValueTF {#1}
77     { \zhnum_counter:n }
78     { \zhnumwithoptions {#1} }
79     {#2}
80 }
81 \NewDocumentCommand \zhnumwithoptions { +m +m }
82 {
83     \group_begin:
84     \keys_set:nn { zhnum / options } {#1}
85     \zhnum_counter:n {#2}
86     \group_end:
87 }

```

(End definition for `\zhnum` and `\zhnumberwithoptions`. These functions are documented on page 4.)

`\zhnum_counter:n` 可以直接通过比较 L^AT_EX 计数器的值来得到符号和绝对值。

```

\zhnum_int:n
88 \cs_new:Npn \zhnum_counter:n #1
89 {
90     \int_if_exist:cTF { c@#1 }
91     { \zhnum_int:c { c@#1 } }
92     { \__zhnum_counter_error:n {#1} }
93 }
94 \cs_new:Npn \__zhnum_counter_error:n #1
95 { \msg_expandable_error:nnn { zhnumber } { not-counter } {#1} }
96 \msg_new:nnn { zhnumber } { not-counter }
97 { `#1' is not a LATEX counter. }
98 \cs_new:Npn \zhnum_int:n #1
99 {
100     \int_compare:nNnTF {#1} > \c_zero
101     { \zhnum_parse_number:f { \int_eval:n {#1} } }
102     {

```

```

103     \int_compare:nNnTF {#1} < \c_zero
104     {
105         \c_zhnum_minus_tl
106         \zhnum_parse_number:f { \int_eval:n { - #1 } }
107     }
108     { \c_zhnum_zero_tl }
109 }
110 }
111 \cs_generate_variant:Nn \zhnum_int:n { c }

```

(End definition for \zhnum_counter:n and \zhnum_int:n.)

\@zhnum 用于支持 \pagenumbering{zhnum}。

```

112 \cs_new_nopar:Npn \@zhnum { \zhnum_int:n }

```

(End definition for \@zhnum.)

\zhnum_integer:n 对整数的处理。这个函数基本抄录自 *l3bigint* 的 `__bingint_read_do:nn`。它可以正确取得符号, 去掉多余的零, 还可以循环展开数字。但它在遇到非数字的时候就停止了循环, 我们可能需要非数字 (例如逗号) 来作为分隔符号。因此对它略作修改, 跳过非数字。

```

113 \cs_new:Npn \zhnum_integer:n #1
114 {
115     \exp_after:wN \__zhnum_read_integer:www
116     \tex_number:D
117     \exp_after:wN \__zhnum_read_sign_loop:N
118     \exp:w \exp_end_continue_f:w \use:n
119     #1 \exp_stop_f: \q_recursion_tail \q_recursion_stop
120     \__zhnum_result:nn { \c_zero } { } ;
121 }
122 \cs_new:Npn \__zhnum_read_sign_loop:N #1
123 {
124     \if:w + \if:w - \exp_not:N #1 + \fi: \exp_not:N #1
125     \exp_after:wN \__zhnum_read_sign_loop:N
126     \exp:w \exp_end_continue_f:w \exp_after:wN \use:n
127     \else:
128     1 \exp_after:wN ;
129     \exp:w \exp_end_continue_f:w
130     \exp_after:wN \__zhnum_read_zeros_loop:N
131     \exp_after:wN #1
132     \fi:
133 }
134 \cs_new:Npn \__zhnum_read_zeros_loop:N #1
135 {
136     \if:w 0 \exp_not:N #1
137     \exp_after:wN \__zhnum_read_zeros_loop:N
138     \exp:w \exp_end_continue_f:w \exp_after:wN \use:n
139     \else:
140     \exp_after:wN \__zhnum_read_abs_loop:Nw
141     \exp_after:wN #1
142     \fi:
143 }

```

(End definition for \zhnum_integer:n.)

__zhnum_read_abs_loop:Nw 当数字很大时, *l3bigint* 的实现会造成 TeX 内存溢出:

! TeX capacity exceeded, sorry [expansion depth=10000].

我们在这里参考 `__tl_act:NNNnn` 的实现对它进行了改进。

```

144 \cs_new:Npn \__zhnum_read_abs_loop:Nw #1#2 \q_recursion_stop
145 {
146     \zhnum_if_digit:NTF #1
147     { \__zhnum_output:nnwnn { + \c_one } #1 }
148     { \quark_if_recursion_tail_stop_do:Nn #1 { \__zhnum_loop_end:wnn } }
149     \exp_after:wN \__zhnum_read_abs_loop:Nw
150     \exp:w \exp_end_continue_f:w \use:n #2 \q_recursion_stop
151 }

```

```

152 \cs_new:Npn \__zhnum_output:nnwnn #1#2#3 \__zhnum_result:nn #4#5
153 { #3 \__zhnum_result:nn { #4#1 } { #5#2 } }
154 \cs_new:Npn \__zhnum_loop_end:wnn #1 \__zhnum_result:nn #2#3
155 { \int_eval:n {#2} ; #3 }

```

(End definition for __zhnum_read_abs_loop:Nw.)

__zhnum_read_integer:www #1 符号, #3 是绝对值, #2 是绝对值的长度。

```

156 \cs_new:Npn \__zhnum_read_integer:www #1 ; #2 ; #3 ;
157 {
158   \int_compare:nNnTF {#2} = \c_zero
159   { \c_zhnum_zero_tl }
160   {
161     \int_compare:nNnF {#1} = \c_one
162     { \c_zhnum_minus_tl }
163     \zhnum_parse_number:nn {#2} {#3}
164   }
165 }

```

(End definition for __zhnum_read_integer:www.)

\zhnum_if_digit:NTF 判断 #1 是否为数字位。

```

166 \cs_new:Npn \zhnum_if_digit:NTF #1
167 {
168   \if_int_compare:w \c_nine < 1 \exp_not:N #1 \exp_stop_f:
169   \exp_after:wN \use_i:nn
170   \else:
171     \exp_after:wN \use_ii:nn
172   \fi:
173 }

```

(End definition for \zhnum_if_digit:NTF.)

\zhnum_parse_number:n
\zhnum_parse_number:nn

```

174 \cs_new:Npn \zhnum_parse_number:n #1
175 { \exp_args:Nf \zhnum_parse_number:nn { \tl_count:n {#1} } {#1} }
176 \cs_new:Npn \zhnum_parse_number:nn #1
177 { \exp_args:Nf \__zhnum_parse_number:nnn { \int_mod:nn {#1} \c_four } {#1} }
178 \cs_new:Npn \__zhnum_parse_number:nnn #1#2
179 {
180   \int_compare:nNnTF {#2} < \c_two
181   { \zhnum_digit_map:n }
182   {
183     \int_compare:nNnTF {#1} = \c_zero
184     { \zhnum_split_number:fn { \int_eval:n { #2 / \c_four - \c_one } } }
185     { \__zhnum_split_number_aux:nnn {#1} {#2} }
186   }
187 }
188 \cs_generate_variant:Nn \zhnum_parse_number:n { f }

```

(End definition for \zhnum_parse_number:n and \zhnum_parse_number:nn.)

__zhnum_split_number_aux:nnn 为了处理的方便, 在整数前面补上适当的 0, 使其位数可以被 4 整除。

```

189 \cs_new:Npn \__zhnum_split_number_aux:nnn #1#2
190 {
191   \exp_after:wN \__zhnum_split_number_aux:wwn
192   \tex_number:D \int_div_truncate:nn {#2} \c_four
193   \if_case:w #1 \exp_stop_f:
194     \or: \exp_after:wN \use:n
195     \or: \exp_after:wN \use_iii:nnn
196     \or: \exp_after:wN \use_i:nnn
197   \fi:
198   { \exp_stop_f: ; 0 } 0 0 ;
199 }
200 \cs_new:Npn \__zhnum_split_number_aux:wwn #1 ; #2 ; #3
201 { \zhnum_split_number:nn {#1} { #2#3 } }

```


(End definition for _zhnum_split_number_aux:nnn.)

\zhnum_split_number:nn 最后加入的 \q_recursion_tail 是停止递归的标志,而 \q_nil 用于占位。

```
202 \cs_new:Npn \zhnum_split_number:nn #1#2
203 {
204   \zhnum_split_number:NNnNNNNw \c_true_bool \c_true_bool {#1}
205   #2 \q_recursion_tail \q_nil \q_nil \q_nil \q_recursion_stop
206 }
207 \cs_generate_variant:Nn \zhnum_split_number:nn { f }
```

(End definition for \zhnum_split_number:nn.)

zhnum_split_number:NNnNNNNw 将输入的整数由高位到低位,以四位为一段进行处理。

```
208 \cs_new:Npn \zhnum_split_number:NNnNNNNw #1#2#3#4#5#6#7
209 {
210   \quark_if_recursion_tail_stop:N #4
211   \int_compare:nNnTF { #4#5#6#7 } = \c_zero
212     { \use_i:nn }
213     {
214       \bool_if:NF #1 { \c__zhnum_zero_tl }
215       \zhnum_process_number:NNNNNN #4#5#6#7#1#2
216       \zhnum_scale_map:n {#3}
217       \int_compare:nNnTF {#7} = \c_zero
218     }
219     { \zhnum_split_number:NNfNNNNw \c_false_bool \c_true_bool }
220     { \zhnum_split_number:NNfNNNNw \c_true_bool \c_false_bool }
221     { \int_eval:n { #3 - \c_one } }
222   }
223 \cs_generate_variant:Nn \zhnum_split_number:NNnNNNNw { NNf }
```

(End definition for \zhnum_split_number:NNnNNNNw.)

zhnum_process_number:NNNNNN 对四位数字按情况进行处理。

```
224 \cs_new:Npn \zhnum_process_number:NNNNNN #1#2#3#4#5#6
225 {
226   \int_compare:nNnTF {#1} = \c_zero
227     { \bool_if:NF #6 { \c__zhnum_zero_tl } }
228     { \zhnum_digit_map:n {#1} \c__zhnum_thousand_tl }
229   \int_compare:nNnTF {#2} = \c_zero
230     { \int_compare:nNnF { #1 * (#3#4) } = \c_zero { \c__zhnum_zero_tl } }
231     {
232       \bool_if:nTF
233         { \l_zhnum_ancient_bool && \int_compare_p:nNn {#2} = \c_two }
234         { \zhnum_digit_map:n { #2 00 } }
235         { \zhnum_digit_map:n {#2} \c__zhnum_hundred_tl }
236     }
237   \int_compare:nNnTF {#3} = \c_zero
238     { \int_compare:nNnF { #2 * #4 } = \c_zero { \c__zhnum_zero_tl } }
239     {
240       \bool_if:nF
241       {
242         \int_compare_p:nNn {#3} = \c_one &&
243         \int_compare_p:nNn {#1#2} = \c_zero && #6 && #5
244       }
245       {
246         \bool_if:nTF
247         {
248           \l_zhnum_ancient_bool &&
249           ( \int_compare_p:nNn {#3} = \c_two ||
250             \int_compare_p:nNn {#3} = \c_three ||
251             \int_compare_p:nNn {#3} = \c_four )
252         }
253         { \zhnum_digit_map:n { #3 0 } \use_none:n }
254         { \zhnum_digit_map:n {#3} }
255       }
256     }
257   \c__zhnum_ten_tl
```

```

257     }
258     \int_compare:nNnF {#4} = \c_zero { \zhnum_digit_map:n {#4} }
259 }

```

(End definition for \zhnum_process_number:NNNNNN.)

\zhdig 用于将 L^AT_EX 计数器按中文数字串输出。

```

260 \DeclareExpandableDocumentCommand \zhdig { +o +m }
261 {
262     \IfNoValueTF {#1}
263     { \zhnum_digits_counter:n }
264     { \zhdigwithoptions {#1} }
265     {#2}
266 }
267 \NewDocumentCommand \zhdigwithoptions { +m +m }
268 {
269     \group_begin:
270     \keys_set:nn { zhnum / options } {#1}
271     \zhnum_digits_counter:n #1 {#2}
272     \group_end:
273 }
274 \cs_new:Npn \zhnum_digits_counter:n #1
275 {
276     \int_if_exist:cTF { c@#1 }
277     { \zhnum_digits_null:v { c@#1 } }
278     { \__zhnum_counter_error:n {#1} }
279 }

```

(End definition for \zhdig. This function is documented on page 4.)

\@zhdig 用于支持 \pagenumbering{zhdig}。

```

280 \cs_new_nopar:Npn \@zhdig #1 { \zhnum_digits_null:f { \int_eval:n {#1} } }

```

(End definition for \@zhdig.)

\zhdigits 将输入的数字输出为中文数字串输出。

```

\zhdigitswithoptions 281 \DeclareExpandableDocumentCommand \zhdigits { +s +o +m }
282 {
283     \IfNoValueTF {#2}
284     { \zhnum_digits:Nn #1 }
285     { \zhdigitswithoptions {#1} {#2} }
286     {#3}
287 }
288 \NewDocumentCommand \zhdigitswithoptions { +m +m +m }
289 {
290     \group_begin:
291     \keys_set:nn { zhnum / options } {#2}
292     \zhnum_digits:Nn #1 {#3}
293     \group_end:
294 }

```

(End definition for \zhdigits and \zhdigitswithoptions. These functions are documented on page 4.)

\zhnum_digits_zero:n 快捷方式。

```

\zhnum_digits_null:n 295 \cs_new_nopar:Npn \zhnum_digits_zero:n
296 { \zhnum_digits:Nn \BooleanTrue }
297 \cs_new_nopar:Npn \zhnum_digits_null:n
298 { \zhnum_digits:Nn \BooleanFalse }
299 \cs_generate_variant:Nn \zhnum_digits_null:n { V , v , f }

```

(End definition for \zhnum_digits_zero:n and \zhnum_digits_null:n.)

`\zhnum_digits:Nn` 与 `\zhnum_integer:n` 类似,但不用去掉多余的零。

```

300 \cs_new:Npn \zhnum_digits:Nn #1#2
301 {
302   \exp_after:wN \__zhnum_read_digits:w
303   \tex_number:D
304   \exp_after:wN \__zhnum_read_sign_loop:NN \exp_after:wN #1
305   \exp:w \exp_end_continue_f:w \use:n
306   #2 \exp_stop_f: \q_recursion_tail \q_recursion_stop
307 }
308 \cs_new:Npn \__zhnum_read_sign_loop:NN #1#2
309 {
310   \if:w + \if:w - \exp_not:N #2 + \fi: \exp_not:N #2
311   \exp_after:wN \__zhnum_read_sign_loop:NN \exp_after:wN #1
312   \exp:w \exp_end_continue_f:w \exp_after:wN \use:n
313   \else:
314     1 \exp_after:wN ;
315     \exp_after:wN \__zhnum_read_digits_loop:NN
316     \exp_after:wN #1
317     \exp_after:wN #2
318   \fi:
319 }
320 \cs_new:Npn \__zhnum_read_digits_loop:NN #1#2
321 {
322   \zhnum_if_digit:NTF #2
323   { \__zhnum_output_digits:NN #1#2 }
324   {
325     \quark_if_recursion_tail_stop:N #2
326     \if:w .\exp_not:N #2 \exp_after:wN \c__zhnum_dot_tl \fi:
327   }
328   \exp_after:wN \__zhnum_read_digits_loop:NN \exp_after:wN #1
329   \exp:w \exp_end_continue_f:w \use:n
330 }
331 \cs_new:Npn \__zhnum_read_digits:w #1 ;
332 {
333   \int_compare:nNnF {#1} = \c_one
334   { \c__zhnum_minus_tl }
335 }
336 \cs_new:Npn \__zhnum_output_digits:NN #1#2
337 {
338   \cs:w
339   c__zhnum_
340   \if_int_compare:w #2 = \c_zero
341   \IfBooleanTF #1 { zero } { null }
342   \else:
343     #2
344   \fi:
345   _tl
346   \cs_end:
347 }

```

(End definition for `\zhnum_digits:Nn`.)

`\zhdate` 输出中文日期。

```

348 \DeclareExpandableDocumentCommand \zhdate { +s +m }
349 {
350   \__zhnum_date:www #2 \q_stop
351   \IfBooleanT #1
352   { \__zhnum_week_day:www #2 \q_stop }
353 }
354 \cs_new:Npn \__zhnum_date:www #1/#2/#3 \q_stop
355 { \__zhnum_date_aux:nnn {#1} {#2} {#3} }

```

(End definition for `\zhdate`. This function is documented on page 2.)

`\zhtoday` 输出当天日期。

```

356 \cs_new_nopar:Npn \zhtoday
357 { \__zhnum_date_aux:Vnn \tex_year:D \tex_month:D \tex_day:D }

```

(End definition for \zhtoday. This function is documented on page 2.)

_zhnum_date_aux:nnn

```

358 \cs_new_nopar:Npn \_zhnum_date_aux:nnn
359 {
360   \bool_if:NTF \l_zhnum_time_bool
361     { \_zhnum_date_aux:NNnnnn \zhnum_digits_null:n \zhnum_int:n { } }
362     { \_zhnum_date_aux:Nnnnn \int_to_arabic:n { \l_zhnum_arabic_sep_tl } }
363 }
364 \cs_new:Npn \_zhnum_date_aux:Nnnnn #1
365 { \_zhnum_date_aux:NNnnnn #1#1 }
366 \cs_new:Npn \_zhnum_date_aux:NNnnnn #1#2#3#4#5#6
367 {
368   #1 {#4} #3 \c_zhnum_year_tl #3
369   #2 {#5} #3 \c_zhnum_month_tl #3
370   #2 {#6} #3 \c_zhnum_day_tl
371 }
372 \cs_generate_variant:Nn \_zhnum_date_aux:nnn { V }

```

(End definition for _zhnum_date_aux:nnn. This function is documented on page ??.)

\zhweekday 输出星期

```

373 \cs_new:Npn \zhweekday #1
374 { \_zhnum_week_day:www #1 \q_stop }

```

(End definition for \zhweekday. This function is documented on page 2.)

_zhnum_week_day:www

用 Zeller 公式计算的结果 h 与实际星期的关系是 $d = h + 5 \pmod{7} + 1$ 。

```

375 \cs_new:Npn \_zhnum_week_day:www #1/#2/#3 \q_stop
376 {
377   \if_case:w \zhnum_Zeller:nnn {#1} {#2} {#3} \exp_stop_f:
378     \c_zhnum_sat_tl
379     \or: \c_zhnum_sun_tl
380     \or: \c_zhnum_mon_tl
381     \or: \c_zhnum_tue_tl
382     \or: \c_zhnum_wed_tl
383     \or: \c_zhnum_thu_tl
384     \or: \c_zhnum_fri_tl
385   \fi:
386 }

```

(End definition for _zhnum_week_day:www.)

\zhnum_Zeller:nnn
 \zhnum_Zeller_aux:Nnnn
 \zhnum_two_digits:n

用 Zeller 公式¹ 计算星期几。

```

387 \cs_new:Npn \zhnum_Zeller:nnn #1#2#3
388 {
389   \int_compare:nNnTF
390     { #1 \zhnum_two_digits:n {#2} \zhnum_two_digits:n {#3} } > { 1582 10 04 }
391     { \_zhnum_Zeller_aux:Nnnn \zhnum_Zeller_Gregorian:nnn }
392     { \_zhnum_Zeller_aux:Nnnn \zhnum_Zeller_Julian:nnn }
393   {#1} {#2} {#3}
394 }
395 \cs_new:Npn \_zhnum_Zeller_aux:Nnnn #1#2#3#4
396 {
397   \int_compare:nNnTF {#3} < \c_three
398     { #1 { #2 - \c_one } { #3 + \c_twelve } {#4} }
399     { #1 {#2} {#3} {#4} }
400 }
401 \cs_new:Npn \zhnum_two_digits:n #1
402 {
403   \int_compare:nNnT {#1} < \c_ten { 0 }
404   \int_eval:n {#1}
405 }

```

¹http://en.wikipedia.org/wiki/Zeller's_congruence

(End definition for `\zhnum_Zeller:nnn`, `\zhnum_Zeller_aux:Nnnn`, and `\zhnum_two_digits:n`.)

`\zhnum_Zeller_Gregorian:nnn` 格里历(1582 年 10 月 15 日及以后)的计算公式

$$h = \left(q + \left\lfloor \frac{26(m+1)}{10} \right\rfloor + Y + \left\lfloor \frac{Y}{4} \right\rfloor + 6 \left\lfloor \frac{Y}{100} \right\rfloor + \left\lfloor \frac{Y}{400} \right\rfloor \right) \pmod{7}$$

其中 Y 为年, m 为月, q 为日; 若 $m = 1, 2$, 则令 $m += 12$, 同时 $Y --$ 。

```

406 \cs_new:Npn \zhnum_Zeller_Gregorian:nnn #1#2#3
407 {
408   \int_mod:nn
409   {
410     (#3)
411     + \int_div_truncate:nn { 26 * ( #2 + \c_one ) } \c_ten
412     + (#1)
413     + \int_div_truncate:nn {#1} \c_four
414     + \c_six * \int_div_truncate:nn {#1} \c_one_hundred
415     + \int_div_truncate:nn {#1} { 400 }
416   }
417   { \c_seven }
418 }

```

(End definition for `\zhnum_Zeller_Gregorian:nnn`.)

`\zhnum_Zeller_Julian:nnn` 儒略历(1582 年 10 月 4 日及以前)的计算公式

$$h = \left(q + \left\lfloor \frac{26(m+1)}{10} \right\rfloor + Y + \left\lfloor \frac{Y}{4} \right\rfloor + 5 \right) \pmod{7}$$

```

419 \cs_new:Npn \zhnum_Zeller_Julian:nnn #1#2#3
420 {
421   \int_mod:nn
422   {
423     (#3)
424     + \int_div_truncate:nn { 26 * ( #2 + \c_one ) } \c_ten
425     + (#1)
426     + \int_div_truncate:nn {#1} \c_four
427     + \c_five
428   }
429   { \c_seven }
430 }

```

(End definition for `\zhnum_Zeller_Julian:nnn`.)

`\zhptime` 输出时间。

```

431 \cs_new:Npn \zhptime #1
432 { \__zhnum_time:ww #1 \q_stop }
433 \use:x
434 {
435   \cs_new:Npn \exp_not:N \__zhnum_time:ww ##1 \c_colon_str ##2 \exp_not:N \q_stop
436 }
437 { \__zhnum_time_aux:nn {#1} {#2} }

```

(End definition for `\zhptime`. This function is documented on page 2.)

`\zhcurrtime` 输出当前时间。

```

438 \cs_new_nopar:Npn \zhcurrtime
439 {
440   \__zhnum_time_aux:nn
441   { \int_div_truncate:nn \tex_time:D { 60 } }
442   { \int_mod:nn \tex_time:D { 60 } }
443 }

```

(End definition for `\zhcurrtime`. This function is documented on page 2.)

```

\__zhnum_time_aux:nn
\__zhnum_time_aux:Nnnn 444 \cs_new_nopar:Npn \__zhnum_time_aux:nn
445 {
446   \bool_if:NTF \l__zhnum_time_bool
447     { \__zhnum_time_aux:Nnnn \zhnum_int:n { } }
448     { \__zhnum_time_aux:Nnnn \int_to_arabic:n { \l__zhnum_arabic_sep_tl } }
449   }
450 \cs_new:Npn \__zhnum_time_aux:Nnnn #1#2#3#4
451 {
452   #1 {#3} #2 \c__zhnum_hour_tl #2
453   #1 {#4} #2 \c__zhnum_minute_tl
454 }

```

(End definition for __zhnum_time_aux:nn and __zhnum_time_aux:Nnnn.)

\zhnum_digit_map:n 阿拉伯数字与中文数字的映射。

```

455 \cs_new:Npn \zhnum_digit_map:n #1
456 { \use:c { c__zhnum_ #1 _tl } }

```

(End definition for \zhnum_digit_map:n.)

\zhnum_scale_map:n 大数系统的映射。

```

\zhnum_scale_map_loop:n 457 \cs_new:Npn \zhnum_scale_map:n #1
458 {
459   \cs_if_exist_use:cF { c__zhnum_s #1 _tl }
460   { \zhnum_scale_map_hook:n {#1} }
461 }
462 \cs_new:Npn \zhnum_scale_map_loop:n #1
463 { \zhnum_scale_map:n { \int_mod:nn {#1} \l__zhnum_scale_int } }
464 \cs_generate_variant:Nn \zhnum_scale_map:n { f }
465 \int_new:N \l__zhnum_scale_int
466 \int_set_eq:NN \l__zhnum_scale_int \c_eleven
467 \cs_new_eq:NN \zhnum_scale_map_hook:n \zhnum_scale_map_loop:n
468 \tl_const:cn { c__zhnum_s0_tl } { }

```

(End definition for \zhnum_scale_map:n and \zhnum_scale_map_loop:n.)

\zhnumExtendScaleMap 扩展进位系统。

```

469 \NewDocumentCommand \zhnumExtendScaleMap { > { \TrimSpaces } +o +m }
470 {
471   \int_zero:N \l_tmpa_int
472   \clist_map_function:nN {#2} \zhnum_set_scale:n
473   \IfNoValueF {#1}
474   { \cs_set:Npn \zhnum_scale_map_hook:n ##1 {#1} }
475 }

```

(End definition for \zhnumExtendScaleMap. This function is documented on page 3.)

\zhnum_set_scale:n

```

476 \cs_new_protected:Npn \zhnum_set_scale:n #1
477 {
478   \int_incr:N \l_tmpa_int
479   \tl_set:Nx \l_tmpa_tl
480   { c__zhnum_s \int_eval:n { \l_tmpa_int + \c_eleven } _tl }
481   \tl_if_exist:cF { \l_tmpa_tl }
482   { \int_incr:N \l__zhnum_scale_int }
483   \tl_set:cn { \l_tmpa_tl } {#1}
484 }

```

(End definition for \zhnum_set_scale:n.)

\zhnum_ganzhi_normal:nnn 保证干支的参数为正数。

```

485 \cs_new:Npn \zhnum_ganzhi_normal:nnn #1#2#3
486 {
487   \int_compare:nNnF {#1} < \c_one
488   { \cs_if_exist_use:c { c__zhnum_ #2 _ #1 _tl } }
489 }

```

(End definition for \zhnum_ganzhi_normal:nnn.)

\zhnum_ganzhi_cyclic:nnn 对超出范围的数字取模, 参数 0 的结果是空值。

```

\__zhnum_ganzhi_cyclic_mod:nnnn 490 \cs_new:Npn \zhnum_ganzhi_cyclic:nnn #1#2#3
491 {
492   \int_compare:nNnF {#1} = \c_zero
493   {
494     \cs_if_exist_use:cF { c__zhnum_ #2 _ #1 _t1 }
495     {
496       \__zhnum_ganzhi_cyclic_mod:fnnn
497       { \int_mod:nn {#1} {#3} } {#1} {#2} {#3}
498     }
499   }
500 }
501 \cs_new:Npn \__zhnum_ganzhi_cyclic_mod:nnnn #1#2#3#4
502 {
503   \int_compare:nNnTF {#2} > \c_zero
504   { \use:c { c__zhnum_ #3 _ #1 _t1 } }
505   {
506     \int_compare:nNnTF {#1} = \c_zero
507     { \use:c { c__zhnum_ #3 _ 1 _t1 } }
508     { \use:c { c__zhnum_ #3 _ \int_eval:n { #1 + #4 + 1 } _t1 } }
509   }
510 }
511 \cs_generate_variant:Nn \__zhnum_ganzhi_cyclic_mod:nnnn { f }

```

(End definition for \zhnum_ganzhi_cyclic:nnn.)

\zhnum_ganzhi:nnn 默认不对超出范围的数字取模。

```

512 \cs_new_eq:NN \zhnum_ganzhi:nnn \zhnum_ganzhi_normal:nnn
513 \cs_generate_variant:Nn \zhnum_ganzhi:nnn { f }

```

(End definition for \zhnum_ganzhi:nnn.)

\zhtiangan 天干。

```

514 \cs_new:Npn \zhtiangan #1
515 { \zhnum_ganzhi:fnn { \int_eval:n {#1} } { tiangan } { 10 } }

```

(End definition for \zhtiangan. This function is documented on page 2.)

\zhdizhi 地支。

```

516 \cs_new:Npn \zhdizhi #1
517 { \zhnum_ganzhi:fnn { \int_eval:n {#1} } { dizhi } { 12 } }

```

(End definition for \zhdizhi. This function is documented on page 2.)

\zhganzhi 干支。

```

518 \cs_new:Npn \zhganzhi #1
519 { \zhnum_ganzhi:fnn { \int_eval:n {#1} } { ganzhi } { 60 } }

```

(End definition for \zhganzhi. This function is documented on page 2.)

\zhganzhinian 干支纪年。

```

520 \cs_new:Npn \zhganzhinian #1
521 { \zhnum_ganzhi_nian:f { \int_eval:n {#1} } }

```

(End definition for \zhganzhinian. This function is documented on page 3.)

\zhnum_ganzhi_nian:n 干支纪年。公元元年是 \zhganzhi{58}。

```

522 \cs_new:Npn \zhnum_ganzhi_nian:n #1
523 {
524   \int_compare:nNnTF {#1} > \c_zero
525   { \use:c { c__zhnum_ganzhi_ \int_mod:nn { #1 + 57 } { 60 } _t1 } }
526   {
527     \int_compare:nNnF {#1} = \c_zero

```

```

528         {
529             \use:c
530             {
531                 c__zhnum_ganzhi_
532                 \int_eval:n { \int_mod:nn { #1 - 2 } { 60 } + 60 }
533                 _tl
534             }
535         }
536     }
537 }
538 \cs_generate_variant:Nn \zhnum_ganzhi_nian:n { f }

```

(End definition for \zhnum_ganzhi_nian:n)

根据需要设置中文阿拉伯数字。

```

539 \group_begin:
540 \tl_set:Nn \l_tmpa_tl
541 {
542     - .tl_set:N = \l__zhnum_minus_tl ,
543     -0 .tl_set:N = \l__zhnum_null_tl ,
544 }
545 \tl_put_right:Nx \l_tmpa_tl
546 {
547     E2 .tl_set:N = \exp_not:c { l__zhnum_ 100 _tl } ,
548     E3 .tl_set:N = \exp_not:c { l__zhnum_ 1000 _tl } ,
549     FE2 .tl_set:N = \exp_not:c { l__zhnum_financial_ 100 _tl } ,
550     FE3 .tl_set:N = \exp_not:c { l__zhnum_financial_ 1000 _tl } ,
551     D11 .tl_set:N = \exp_not:c { l__zhnum_dizhi_ 11 _tl } ,
552     D12 .tl_set:N = \exp_not:c { l__zhnum_dizhi_ 12 _tl } ,
553     E44 .tl_set:N = \exp_not:c { l__zhnum_ s11 _tl } ,
554 }
555 \int_step_inline:nnnn { 1 } { 1 } { 10 }
556 {
557     \tl_put_right:Nx \l_tmpa_tl
558     {
559         #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } ,
560         F#1 .tl_set:N = \exp_not:c { l__zhnum_financial_ #1 _tl } ,
561         T#1 .tl_set:N = \exp_not:c { l__zhnum_tiangang_ #1 _tl } ,
562         D#1 .tl_set:N = \exp_not:c { l__zhnum_dizhi_ #1 _tl } ,
563         GZ#1 .tl_set:N = \exp_not:c { l__zhnum_ganzhi_ #1 _tl } ,
564         E \int_eval:n { #1 * 4 }
565         .tl_set:N = \exp_not:c { l__zhnum_ s#1 _tl } ,
566     }
567 }
568 \int_step_inline:nnnn { 11 } { 1 } { 60 }
569 {
570     \tl_put_right:Nx \l_tmpa_tl
571     { GZ#1 .tl_set:N = \exp_not:c { l__zhnum_ganzhi_ #1 _tl } , }
572 }
573 \clist_map_inline:nn { 0 , 100 , 1000 }
574 {
575     \tl_put_right:Nx \l_tmpa_tl
576     {
577         #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } ,
578         F#1 .tl_set:N = \exp_not:c { l__zhnum_financial_ #1 _tl } ,
579     }
580 }
581 \clist_map_inline:nn { 20 , 30 , 40 , 200 }
582 {
583     \tl_put_right:Nx \l_tmpa_tl
584     { #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } , }
585 }
586 \clist_map_inline:nn
587 {
588     dot , and , parts , year , month , day , weekday , hour , minute
589     mon , tue , wed , thu , fri , sat , sun
590 }
591 {

```



```

592         \tl_put_right:Nx \l_tmpa_tl
593         { #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } , }
594     }
595 \use:x
596 {
597     \group_end:
598     \keys_define:nn { zhnum / options } { \exp_not:o \l_tmpa_tl }
599 }

```

将配置文件中的中文数字保存到 prop 变量中。

```

\zhnum_set_digits_map:nn 600 \cs_new_protected:Npn \zhnum_set_digits_map:nn #1#2
\zhnum_set_digits_map:nnn 601 { \prop_put:Nnn \l__zhnum_cfg_map_prop {#1} {#2} }
\zhnum_set_financial_map:nn 602 \cs_new_protected:Npn \zhnum_set_digits_map:nnn #1#2#3
\zhnum_set_financial_map:nnn 603 {
\zhnum_set_tiangang_map:nn 604     \prop_put_if_new:Nnn \l__zhnum_cfg_map_prop {#1} {#3}
\zhnum_set_dizhi_map:nn 605     \prop_put:Nnn \l__zhnum_cfg_map_var_prop {#1_#2} {#3}
\l__zhnum_cfg_map_prop 606 }
\l__zhnum_cfg_map_var_prop 607 \cs_new_protected:Npn \zhnum_set_financial_map:nn #1#2
\l__zhnum_cfg_map_finan_prop 608 { \prop_put:Nnn \l__zhnum_cfg_map_finan_prop {#1} {#2} }
\l__zhnum_cfg_map_ganzhi_prop 609 \cs_new_protected:Npn \zhnum_set_financial_map:nnn #1#2#3
610 {
611     \prop_put_if_new:Nnn \l__zhnum_cfg_map_finan_prop {#1} {#3}
612     \prop_put:Nnn \l__zhnum_cfg_map_var_prop { financial_#1_#2 } {#3}
613 }
614 \cs_new_protected:Npn \zhnum_set_tiangang_map:nn #1#2
615 { \prop_put:Nnn \l__zhnum_cfg_map_ganzhi_prop { tiangan_#1 } {#2} }
616 \cs_new_protected:Npn \zhnum_set_dizhi_map:nn #1#2
617 { \prop_put:Nnn \l__zhnum_cfg_map_ganzhi_prop { dizhi_#1 } {#2} }
618 \prop_new:N \l__zhnum_cfg_map_prop
619 \prop_new:N \l__zhnum_cfg_map_var_prop
620 \prop_new:N \l__zhnum_cfg_map_finan_prop
621 \prop_new:N \l__zhnum_cfg_map_ganzhi_prop

```

(End definition for \zhnum_set_digits_map:nn and others.)

将 prop 表转化到单独的 tl 变量。

```

\zhnum_parse_config: 622 \cs_new_protected_nopar:Npn \zhnum_parse_config:
\zhnum_check_simp:nn 623 {
\zhnum_check_financial:nn 624     \prop_map_function:NN \l__zhnum_cfg_map_prop \zhnum_check_simp:nn
\zhnum_set_zero: 625     \prop_map_function:NN \l__zhnum_cfg_map_ganzhi_prop \zhnum_assgin_ganzhi:nn
\zhnum_set_week_day: 626     \zhnum_set_zero:
627     \zhnum_set_week_day:
628     \bool_if:NF \l__zhnum_reset_bool
629     {
630         \zhnum_assgin_const:
631         \bool_set_true:N \l__zhnum_reset_bool
632     }
633 }
634 \cs_new_protected:Npn \zhnum_check_simp:nn #1#2
635 {
636     \__zhnum_check_simp_aux:nn {#2} {#1}
637     \prop_get:NnNT \l__zhnum_cfg_map_finan_prop {#1} \l_tmpa_tl
638     { \exp_args:No \__zhnum_check_simp_aux:nn { \l_tmpa_tl } { financial_ #1 } }
639 }
640 \cs_new_protected:Npn \__zhnum_check_simp_aux:nn #1#2
641 {
642     \prop_get:NnNTF \l__zhnum_cfg_map_var_prop { #2 _trad } \l_tmpa_tl
643     {
644         \prop_get:NnNF \l__zhnum_cfg_map_var_prop { #2 _simp } \l_tmpb_tl
645         { \tl_set:Nn \l_tmpb_tl {#1} }
646         \tl_set:cx { l__zhnum_ #2 _tl }
647         {
648             \exp_not:n { \bool_if:NTF \l__zhnum_simp_bool }
649             { \exp_not:o \l_tmpb_tl } { \exp_not:o \l_tmpa_tl }
650         }
651     }

```

```

652     { \tl_set:cn { l__zhnum_ #2 _tl } {#1} }
653   }
654   \cs_new_protected_nopar:Npn \zhnum_assgin_const:
655   {
656     \prop_map_function:NN \l__zhnum_cfg_map_prop \zhnum_check_financial:nn
657     \zhnum_set_alias:
658   }
659   \cs_new_protected:Npn \zhnum_check_financial:nn #1#2
660   {
661     \prop_get:NnNTF \l__zhnum_cfg_map_finan_prop {#1} \l_tmpa_tl
662     {
663       \zhnum_assgin_const_tl:cx { c__zhnum_ #1 _tl }
664       {
665         \exp_not:n { \bool_if:NTF \l__zhnum_normal_bool }
666         { \exp_not:c { l__zhnum_ #1 _tl } }
667         { \exp_not:c { l__zhnum_financial_ #1 _tl } }
668       }
669     }
670     {
671       \zhnum_assgin_const_tl:cx
672       { c__zhnum_ #1 _tl } { \exp_not:c { l__zhnum_ #1 _tl } }
673     }
674   }
675   \cs_new_protected_nopar:Npn \zhnum_set_zero:
676   {
677     \tl_set:cx { l__zhnum_0_tl }
678     {
679       \exp_not:n { \bool_if:NTF \l__zhnum_null_bool }
680       { \exp_not:o \l__zhnum_null_tl } { \exp_not:v { l__zhnum_0_tl } }
681     }
682   }
683   \cs_new_protected_nopar:Npn \zhnum_set_week_day:
684   {
685     \tl_set:Nx \l__zhnum_mon_tl
686     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_1_tl } }
687     \tl_set:Nx \l__zhnum_tue_tl
688     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_2_tl } }
689     \tl_set:Nx \l__zhnum_wed_tl
690     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_3_tl } }
691     \tl_set:Nx \l__zhnum_thu_tl
692     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_4_tl } }
693     \tl_set:Nx \l__zhnum_fri_tl
694     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_5_tl } }
695     \tl_set:Nx \l__zhnum_sat_tl
696     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_6_tl } }
697     \tl_set:Nx \l__zhnum_sun_tl
698     { \exp_not:N \c__zhnum_weekday_tl \exp_not:o \l__zhnum_day_tl }
699   }
700   \clist_map_inline:nn { mon , tue , wed , thu , fri , sat , sun }
701   { \tl_const:cx { c__zhnum_ #1 _tl } { \exp_not:c { l__zhnum_ #1 _tl } } }
702   \cs_new_protected:Npn \zhnum_assgin_ganzhi:nn #1#2
703   { \tl_set:cn { l__zhnum_ #1 _tl } {#2} }
704   \cs_new:Npn \zhnum_zero_mod:nn #1#2
705   { \exp_args:Nf \__zhnum_zero_mod_aux:nn { \int_mod:nn {#1} {#2} } {#2} }
706   \cs_new:Npn \__zhnum_zero_mod_aux:nn #1#2
707   { \int_compare:nNnTF {#1} = \c_zero {#2} {#1} }
708   \int_step_inline:nnnn { 1 } { 1 } { 60 }
709   {
710     \tl_const:cx { c__zhnum_ganzhi_ #1 _tl } { \exp_not:c { l__zhnum_ganzhi_ #1 _tl } }
711     \tl_set:cx { l__zhnum_ganzhi_ #1 _tl }
712     {
713       \exp_not:c { l__zhnum_tiangang_ \zhnum_zero_mod:nn {#1} { 10 } _tl }
714       \exp_not:c { l__zhnum_dizhi_ \zhnum_zero_mod:nn {#1} { 12 } _tl }
715     }
716   }
717   \cs_new_eq:cc { c__zhnum_ganzhi_ 0 _tl } { c__zhnum_ganzhi_ 60 _tl }
718   \cs_new_eq:NN \zhnum_assgin_const_tl:cx \tl_const:cx

```

```

719 \AtEndOfPackage
720 {
721   \prop_map_inline:Nn \l__zhnum_cfg_map_ganzhi_prop
722   { \tl_const:cx { c__zhnum_ #1 _tl } { \exp_not:c { l__zhnum_ #1 _tl } } }
723   \cs_new_eq:cc { c__zhnum_tiangang_ 0 _tl } { c__zhnum_tiangang_ 10 _tl }
724   \cs_new_eq:cc { c__zhnum_dizhi_ 0 _tl } { c__zhnum_dizhi_ 12 _tl }
725   \cs_set_eq:NN \zhnum_assgin_const_tl:cx \tl_set:cx
726 }

```

(End definition for \zhnum_parse_config: and others.)

\zhnum_set_alias: 一些易于使用的别名。

```

727 \cs_new_eq:NN \zhnum_set_alias:NN \cs_new_eq:NN
728 \cs_new_protected_nopar:Npx \zhnum_set_alias:
729 {
730   \zhnum_set_alias:NN \exp_not:N \c__zhnum_zero_tl
731   \exp_not:c { c__zhnum_ 0 _tl }
732   \zhnum_set_alias:NN \exp_not:N \c__zhnum_ten_tl
733   \exp_not:c { c__zhnum_ 10 _tl }
734   \zhnum_set_alias:NN \exp_not:N \c__zhnum_hundred_tl
735   \exp_not:c { c__zhnum_ 100 _tl }
736   \zhnum_set_alias:NN \exp_not:N \c__zhnum_thousand_tl
737   \exp_not:c { c__zhnum_ 1000 _tl }
738 }
739 \AtEndOfPackage
740 { \cs_set_eq:NN \zhnum_set_alias:NN \tl_set_eq:NN }

```

(End definition for \zhnum_set_alias:.)

\zhnum_load_cfg:n 根据选定编码载入配置文件。

```

741 \cs_new_protected:Npn \zhnum_load_cfg:n #1
742 {
743   \zhnum_set_cfg_name:Nn \l__zhnum_cfg_str {#1}
744   \str_if_eq:NNF \l__zhnum_cfg_str \l__zhnum_last_cfg_str
745   { \zhnum_update_cfg:n {#1} }
746   \zhnum_parse_config:
747 }
748 \cs_generate_variant:Nn \zhnum_load_cfg:n { o }
749 \cs_new_protected:Npn \zhnum_update_cfg:n #1
750 {
751   \prop_if_exist:cTF { g__zhnum_cfg_ \l__zhnum_cfg_str _prop }
752   { \str_set_eq:NN \l__zhnum_last_cfg_str \l__zhnum_cfg_str }
753   { \zhnum_input_cfg:n {#1} }
754   \__zhnum_update_cfg_prop:N \prop_set_eq:Nc
755 }
756 \cs_new_protected:Npn \zhnum_input_cfg:n #1
757 {
758   \file_if_exist_input:nTF { zhnumber - #1 .cfg }
759   {
760     \bool_set_false:N \l__zhnum_reset_bool
761     \__zhnum_update_cfg_prop:N \__zhnum_prop_initial:Nn
762     \group_begin:
763     \zhnum_set_catcode:
764   }
765   {
766     \msg_error:nxx { zhnumber } { file-not-found } {#1}
767     \use_none:nnn
768   }
769   \__zhnum_update_cfg_prop:N \__zhnum_prop_gset_eq:Nn
770   \group_end:
771 }
772 \cs_new_protected:Npn \__zhnum_update_cfg_prop:N #1
773 {
774   #1 \l__zhnum_cfg_map_prop { g__zhnum_cfg_ \l__zhnum_cfg_str _prop }
775   #1 \l__zhnum_cfg_map_var_prop { g__zhnum_cfg_var_ \l__zhnum_cfg_str _prop }
776   #1 \l__zhnum_cfg_map_finan_prop { g__zhnum_cfg_finan_ \l__zhnum_cfg_str _prop }
777   #1 \l__zhnum_cfg_map_ganzhi_prop { g__zhnum_cfg_ganzhi_ \l__zhnum_cfg_str _prop }

```

```

778 }
779 \cs_new_protected:Npn \__zhnum_prop_initial:Nn #1#2
780 {
781   \prop_clear:N #1
782   \prop_new:c {#2}
783 }
784 \cs_new_protected:Npn \__zhnum_prop_gset_eq:Nn #1#2
785 { \prop_gset_eq:cN {#2} #1 }
786 \str_new:N \l__zhnum_cfg_str
787 \str_new:N \l__zhnum_last_cfg_str
788 \bool_new:N \l__zhnum_reset_bool
789 \msg_new:nnnn { zhnumber } { file-not-found }
790 { File~`#1'~not~found. }
791 {
792   The~requested~file~could~not~be~found~in~the~current~directory,~
793   in~the~TeX~search~path~or~in~the~LaTeX~search~path.
794 }

```

(End definition for \zhnum_load_cfg:n.)

\zhnum_if_unicode_engine_p: 使用 upTeX 的时候,也不必将汉字的首字符设置为活动字符。判断 ~~~~0021 是否为单个记号的办法对 upTeX 不适用。

\zhnum_if_unicode_engine:TF

```

795 \bool_if:nTF
796 {
797   \sys_if_engine_xetex_p: ||
798   \sys_if_engine_luatex_p: ||
799   \sys_if_engine_uptex_p:
800 }
801 {
802   \cs_new_eq:NN \zhnum_if_unicode_engine_p: \c_true_bool
803   \cs_new_eq:NN \zhnum_if_unicode_engine:TF \use_i:nn
804 }
805 {
806   \cs_new_eq:NN \zhnum_if_unicode_engine_p: \c_false_bool
807   \cs_new_eq:NN \zhnum_if_unicode_engine:TF \use_ii:nn
808 }

```

(End definition for \zhnum_if_unicode_engine:TF.)

\zhnum_set_catcode: 设置与恢复配置文件前后的 catcode。pdfLaTeX 需要将汉字的首字节设置为活动字符。

\zhnum_set_cfg_name:Nn

\zhnum_reset_config:

```

809 \if_predicate:w \zhnum_if_unicode_engine_p:
810 \cs_new_eq:NN \zhnum_set_catcode: \prg_do_nothing:
811 \cs_new_protected:Npn \zhnum_set_cfg_name:Nn #1#2
812 {
813   \str_set:Nx \l__zhnum_encoding_str {#2}
814   \str_set_eq:NN #1 \l__zhnum_encoding_str
815 }
816 \cs_new_eq:NN \zhnum_reset_config: \zhnum_parse_config:
817 \else:
818 \cs_new_protected_nopar:Npn \zhnum_set_catcode:
819 { \bool_if:NT \l__zhnum_active_char_bool { \zhnum_set_active: } }
820 \cs_new_protected_nopar:Npn \zhnum_set_active:
821 {
822   \str_case:onTF { \l__zhnum_encoding_str }
823   {
824     { gbk } { \int_set:Nn \l__zhnum_byte_min_int { "81 } }
825     { big5 } { \int_set:Nn \l__zhnum_byte_min_int { "A1 } }
826   }
827   { \int_set:Nn \l__zhnum_byte_max_int { "FE } }
828   {
829     \int_set:Nn \l__zhnum_byte_min_int { "E0 }
830     \int_set:Nn \l__zhnum_byte_max_int { "EF }
831   }
832   \int_step_function:nnnN
833   { \l__zhnum_byte_min_int } { \c_one }
834   { \l__zhnum_byte_max_int } \char_set_catcode_active:n

```

```

835     }
836     \int_new:N \l__zhnum_byte_min_int
837     \int_new:N \l__zhnum_byte_max_int
838     \cs_new_protected:Npn \zhnum_set_cfg_name:Nn #1#2
839     {
840         \str_set:Nx \l__zhnum_encoding_str {#2}
841         \str_set:Nx #1
842         {
843             \l__zhnum_encoding_str
844             \bool_if:NT \l__zhnum_active_char_bool { _active }
845         }
846     }
847     \cs_new_protected_nopar:Npn \zhnum_reset_config:
848     { \zhnum_load_cfg:o { \l__zhnum_encoding_str } }
849     \bool_new:N \l__zhnum_active_char_bool
850     \bool_set_true:N \l__zhnum_active_char_bool
851     \fi:

```

(End definition for \zhnum_set_catcode:, \zhnum_set_cfg_name:Nn, and \zhnum_reset_config:.)

encoding 宏包设置选项。

```

style 852 \keys_define:nn { zhnum / options }
null 853 {
reset 854     encoding .choices:nn =
855         { UTF8 , GBK , Big5 }
856         {
857             \str_set:Nx \l__zhnum_encoding_str
858             { \str_fold_case:V \l_keys_choice_tl }
859             \zhnum_load_cfg:o { \l__zhnum_encoding_str }
860         } ,
861     encoding .default:n = { GBK } ,
862     encoding / Big5 .meta:n = { encoding = Big5 } ,
863     encoding / unknown .code:n =
864     { \msg_error:nnn { zhnumber } { encoding-invalid } {#1} } ,
865     style .multichoice: ,
866     style / Normal .code:n =
867     {
868         \bool_set_false:N \l__zhnum_ancient_bool
869         \bool_set_true:N \l__zhnum_normal_bool
870     } ,
871     style / Financial .code:n =
872     {
873         \bool_set_false:N \l__zhnum_ancient_bool
874         \bool_set_false:N \l__zhnum_normal_bool
875     } ,
876     style / Ancient .code:n =
877     {
878         \bool_set_true:N \l__zhnum_ancient_bool
879         \bool_set_true:N \l__zhnum_normal_bool
880     } ,
881     style / Simplified .code:n = { \bool_set_true:N \l__zhnum_simp_bool } ,
882     style / Traditional .code:n = { \bool_set_false:N \l__zhnum_simp_bool } ,
883     style .default:n = { Normal , Simplified } ,
884     null .bool_set:N = \l__zhnum_null_bool ,
885     time .choice: ,
886     time / Chinese .code:n = { \bool_set_true:N \l__zhnum_time_bool } ,
887     time / Arabic .code:n = { \bool_set_false:N \l__zhnum_time_bool } ,
888     time .default:n = { Arabic } ,
889     reset .code:n = { \zhnum_reset_config: } ,
890     activechar .bool_set:N = \l__zhnum_active_char_bool ,
891     ganzhi-cyclic .choice: ,
892     ganzhi-cyclic / true .code:n =
893     { \cs_set_eq:NN \zhnum_ganzhi:nnn \zhnum_ganzhi_cyclic:nnn } ,
894     ganzhi-cyclic / false.code:n =
895     { \cs_set_eq:NN \zhnum_ganzhi:nnn \zhnum_ganzhi_normal:nnn } ,
896     ganzhi-cyclic .default:n = { true } ,
897     arabicsep .tl_set:N = \l__zhnum_arabic_sep_tl

```

```

898 }
899 \str_new:N \l__zhnum_encoding_str
900 \msg_new:nnnn { zhnumber } { encoding-invalid }
901 { The~encoding~`#1'~is~invalid. }
902 { Available~encodings~are~`UTF8',~`GBK'~and~`Big5'. }

```

(End definition for encoding and others. These functions are documented on page 1.)

`\zhnumsetup` 在文档中设置 `zhnumber` 的接口。

```

903 \NewDocumentCommand \zhnumsetup { +m }
904 {
905   \keys_set:nn { zhnum / options } {#1}
906   \tex_ignorespaces:D
907 }

```

(End definition for \zhnumsetup. This function is documented on page 3.)

初始化设置和执行宏包选项。

```

908 \keys_set:nn { zhnum / options } { style , time , arabicsep = { ~ } }
909 \ProcessKeysOptions { zhnum / options }

```

如果没有选定编码,则根据引擎自动设置编码。

```

910 \str_if_empty:NT \l__zhnum_encoding_str
911 {
912   \zhnum_if_unicode_engine:TF
913   { \keys_set:nn { zhnum / options } { encoding = UTF8 } }
914   { \keys_set:nn { zhnum / options } { encoding = GBK } }
915 }
916 </package>

```

第 4 节 中文数字配置文件

```

917 <*config>
918 <!*big5>
919 \zhnum_set_digits_map:nnn { minus } { simp } { 负 }
920 \zhnum_set_digits_map:nnn { minus } { trad } { 負 }
921 </!big5>
922 <*big5>
923 \zhnum_set_digits_map:nn { minus } { 負 }
924 </big5>
925 \zhnum_set_digits_map:nn { 0 } { 零 }
926 <!*big5>
927 \zhnum_set_digits_map:nn { null } { 〇 }
928 </!big5>
929 <*big5>
930 \zhnum_set_digits_map:nn { null } { 〇 }
931 </big5>
932 \zhnum_set_digits_map:nn { 1 } { 一 }
933 \zhnum_set_digits_map:nn { 2 } { 二 }
934 \zhnum_set_digits_map:nn { 3 } { 三 }
935 \zhnum_set_digits_map:nn { 4 } { 四 }
936 \zhnum_set_digits_map:nn { 5 } { 五 }
937 \zhnum_set_digits_map:nn { 6 } { 六 }
938 \zhnum_set_digits_map:nn { 7 } { 七 }
939 \zhnum_set_digits_map:nn { 8 } { 八 }
940 \zhnum_set_digits_map:nn { 9 } { 九 }
941 \zhnum_set_digits_map:nn { 10 } { 十 }
942 \zhnum_set_digits_map:nn { 100 } { 百 }
943 \zhnum_set_digits_map:nn { 1000 } { 千 }
944 \zhnum_set_digits_map:nn { 20 } { 廿 }
945 \zhnum_set_digits_map:nn { 30 } { 卅 }
946 \zhnum_set_digits_map:nn { 40 } { 卌 }
947 \zhnum_set_digits_map:nn { 200 } { 佰 }
948 <!*big5>
949 \zhnum_set_digits_map:nnn { dot } { simp } { 点 }

```

```

950 \zhnum_set_digits_map:nnn { dot } { trad } { 點 }
951 </big5>
952 <*big5>
953 \zhnum_set_digits_map:nn { dot } { 點 }
954 </big5>
955 \zhnum_set_digits_map:nn { and } { 又 }
956 \zhnum_set_digits_map:nn { parts } { 分之 }
957 <*!big5>
958 \zhnum_set_digits_map:nnn { s1 } { simp } { 万 }
959 \zhnum_set_digits_map:nnn { s1 } { trad } { 萬 }
960 \zhnum_set_digits_map:nnn { s2 } { simp } { 亿 }
961 \zhnum_set_digits_map:nnn { s2 } { trad } { 億 }
962 </big5>
963 <*big5>
964 \zhnum_set_digits_map:nn { s1 } { 萬 }
965 \zhnum_set_digits_map:nn { s2 } { 億 }
966 </big5>
967 \zhnum_set_digits_map:nn { s3 } { 兆 }
968 \zhnum_set_digits_map:nn { s4 } { 京 }
969 \zhnum_set_digits_map:nn { s5 } { 垓 }
970 \zhnum_set_digits_map:nn { s6 } { 秭 }
971 \zhnum_set_digits_map:nn { s7 } { 穰 }
972 <*!big5>
973 \zhnum_set_digits_map:nnn { s8 } { simp } { 沟 }
974 \zhnum_set_digits_map:nnn { s8 } { trad } { 溝 }
975 \zhnum_set_digits_map:nnn { s9 } { simp } { 涧 }
976 \zhnum_set_digits_map:nnn { s9 } { trad } { 澗 }
977 </big5>
978 <*big5>
979 \zhnum_set_digits_map:nn { s8 } { 溝 }
980 \zhnum_set_digits_map:nn { s9 } { 澗 }
981 </big5>
982 \zhnum_set_digits_map:nn { s10 } { 正 }
983 <*!big5>
984 \zhnum_set_digits_map:nnn { s11 } { simp } { 载 }
985 \zhnum_set_digits_map:nnn { s11 } { trad } { 載 }
986 </big5>
987 <*big5>
988 \zhnum_set_digits_map:nn { s11 } { 載 }
989 </big5>
990 \zhnum_set_digits_map:nn { year } { 年 }
991 \zhnum_set_digits_map:nn { month } { 月 }
992 \zhnum_set_digits_map:nn { day } { 日 }
993 <*!big5>
994 \zhnum_set_digits_map:nnn { hour } { simp } { 时 }
995 \zhnum_set_digits_map:nnn { hour } { trad } { 時 }
996 </big5>
997 <*big5>
998 \zhnum_set_digits_map:nn { hour } { 時 }
999 </big5>
1000 \zhnum_set_digits_map:nn { minute } { 分 }
1001 \zhnum_set_digits_map:nn { weekday } { 星期 }
1002 \zhnum_set_financial_map:nn { null } { 零 }
1003 \zhnum_set_financial_map:nn { 0 } { 零 }
1004 \zhnum_set_financial_map:nn { 1 } { 壹 }
1005 \zhnum_set_financial_map:nn { 2 } { 貳 }
1006 <*!big5>
1007 \zhnum_set_financial_map:nnn { 3 } { simp } { 叁 }
1008 \zhnum_set_financial_map:nnn { 3 } { trad } { 參 }
1009 </big5>
1010 <*big5>
1011 \zhnum_set_financial_map:nn { 3 } { 參 }
1012 </big5>
1013 \zhnum_set_financial_map:nn { 4 } { 肆 }
1014 \zhnum_set_financial_map:nn { 5 } { 伍 }
1015 <*!big5>
1016 \zhnum_set_financial_map:nnn { 6 } { simp } { 陆 }

```

```
1017 \zhnum_set_financial_map:nn { 6 } { trad } { 陸 }
1018 </big5>
1019 <*big5>
1020 \zhnum_set_financial_map:nn { 6 } { 陸 }
1021 </big5>
1022 \zhnum_set_financial_map:nn { 7 } { 柒 }
1023 \zhnum_set_financial_map:nn { 8 } { 捌 }
1024 \zhnum_set_financial_map:nn { 9 } { 玖 }
1025 \zhnum_set_financial_map:nn { 10 } { 拾 }
1026 \zhnum_set_financial_map:nn { 100 } { 佰 }
1027 \zhnum_set_financial_map:nn { 1000 } { 仟 }
1028 \zhnum_set_tiangang_map:nn { 1 } { 甲 }
1029 \zhnum_set_tiangang_map:nn { 2 } { 乙 }
1030 \zhnum_set_tiangang_map:nn { 3 } { 丙 }
1031 \zhnum_set_tiangang_map:nn { 4 } { 丁 }
1032 \zhnum_set_tiangang_map:nn { 5 } { 戊 }
1033 \zhnum_set_tiangang_map:nn { 6 } { 己 }
1034 \zhnum_set_tiangang_map:nn { 7 } { 庚 }
1035 \zhnum_set_tiangang_map:nn { 8 } { 辛 }
1036 \zhnum_set_tiangang_map:nn { 9 } { 壬 }
1037 \zhnum_set_tiangang_map:nn { 10 } { 癸 }
1038 \zhnum_set_dizhi_map:nn { 1 } { 子 }
1039 \zhnum_set_dizhi_map:nn { 2 } { 丑 }
1040 \zhnum_set_dizhi_map:nn { 3 } { 寅 }
1041 \zhnum_set_dizhi_map:nn { 4 } { 卯 }
1042 \zhnum_set_dizhi_map:nn { 5 } { 辰 }
1043 \zhnum_set_dizhi_map:nn { 6 } { 巳 }
1044 \zhnum_set_dizhi_map:nn { 7 } { 午 }
1045 \zhnum_set_dizhi_map:nn { 8 } { 未 }
1046 \zhnum_set_dizhi_map:nn { 9 } { 申 }
1047 \zhnum_set_dizhi_map:nn { 10 } { 酉 }
1048 \zhnum_set_dizhi_map:nn { 11 } { 戌 }
1049 \zhnum_set_dizhi_map:nn { 12 } { 亥 }
1050 </config>
```


代码索引

意大利体的数字表示描述对应索引项的页码;带下划线的数字表示定义对应索引项的代码行号;罗马字体的数字表示使用对应索引项的代码行号。

Symbols	
\\	5, 6, 7
A	
activechar	4
arabicsep	3
\AtEndOfPackage	719, 739
B	
bingint commands:	
_bingint_read_do:nn	7
bool commands:	
_bool_if:Nf	214, 227, 628
_bool_if:nf	240
_bool_if:NT	819, 844
_bool_if:NTF	360, 446, 648, 665, 679
_bool_if:nTF	232, 246, 795
_bool_new:N	788, 849
_bool_set_false:N	760, 868, 873, 874, 882, 887
_bool_set_true:N	631, 850, 869, 878, 879, 881, 886
\BooleanFalse	298
\BooleanTrue	296
C	
char commands:	
_char_set_catcode_active:n	834
clist commands:	
_clist_map_function:nN	472
_clist_map_inline:nn	573, 581, 586, 700
colon commands:	
_c_colon_str	435
cs commands:	
_cs:w	338
_cs_end:	346
_cs_generate_variant:Nn	35, 111, 188, 207, 223, 299, 372, 464, 511, 513, 538, 748
_cs_if_exist_use:c	488
_cs_if_exist_use:cF	459, 494
_cs_new:Npn	27, 29, 36, 42, 61, 68, 88, 94, 98, 113, 122, 134, 144, 152, 154, 156, 166, 174, 176, 178, 189, 200, 202, 208, 224, 274, 300, 308, 320, 331, 336, 354, 364, 366, 373, 375, 387, 395, 401, 406, 419, 431, 435, 450, 455, 457, 462, 485, 490, 501, 514, 516, 518, 520, 522, 704, 706
_cs_new_eq:cc	717, 723, 724
_cs_new_eq:NN	467, 512, 718, 727, 802, 803, 806, 807, 810, 816
_cs_new_nopar:Npn	112, 280, 295, 297, 356, 358, 438, 444
_cs_new_protected:Npn	476, 600, 602, 607, 609, 614, 616, 634, 640, 659, 702, 741, 749, 756, 772, 779, 784, 811, 838
_cs_new_protected_nopar:Npn	622, 654, 675, 683, 818, 820, 847
_cs_new_protected_nopar:Npx	728
_cs_set:Npn	474
_cs_set_eq:NN	725, 740, 893, 895
D	
\DeclareExpandableDocumentCommand	13, 74, 260, 281, 348
E	
eleven commands:	
_c_eleven	466, 480
else commands:	
_else:	127, 139, 170, 313, 342, 817
encoding	1, 21
exp commands:	
_exp:w	118, 126, 129, 138, 150, 305, 312, 329
_exp_after:wN	115, 117, 125, 126, 128, 130, 131, 137, 138, 140, 141, 149, 169, 171, 191, 194, 195, 196, 302, 304, 311, 312, 314, 315, 316, 317, 326, 328
_exp_args:Nf	175, 177, 705
_exp_args:No	638
_exp_end_continue_f:w	118, 126, 129, 138, 150, 305, 312, 329
_exp_not:c	547, 548, 549, 550, 551, 552, 553, 559, 560, 561, 562, 563, 565, 571, 577, 578, 584, 593, 666, 667, 672, 701, 710, 713, 714, 722, 731, 733, 735, 737
_exp_not:N	124, 136, 168, 310, 326, 435, 686, 688, 690, 692, 694, 696, 698, 730, 732, 734, 736
_exp_not:n	648, 665, 679
_exp_not:o	598, 649, 680, 698
_exp_not:v	680, 686, 688, 690, 692, 694, 696
_exp_stop_f:	119, 168, 193, 198, 306, 377
F	
false commands:	
_c_false_bool	219, 220, 806
fi commands:	
_fi:	124, 132, 142, 172, 197, 310, 318, 326, 344, 385, 851
file commands:	
_file_if_exist_input:nTF	758
five commands:	
_c_five	427
four commands:	
_c_four	177, 184, 192, 251, 413, 426
G	
ganzhi-cyclic	4
group commands:	
_group_begin:	22, 83, 269, 290, 539, 762
_group_end:	25, 86, 272, 293, 597, 770
I	
if commands:	
_if:w	124, 136, 310, 326

`\if_case:w` 193, 377
`\if_int_compare:w` 168, 340
`\if_predicate:w` 809
`\IfBooleanT` 351
`\IfBooleanTF` 341
`\IfNoValueF` 473
`\IfNoValueTF` 15, 76, 262, 283

int commands:

`\int_compare:nNnF` . 161, 230, 238, 258, 333, 487, 492, 527
`\int_compare:nNnT` 403
`\int_compare:nNnTF` 100, 103, 158, 180,
 183, 211, 217, 226, 229, 237, 389, 397, 503, 506, 524, 707
`\int_compare_p:nNn` 233, 242, 243, 249, 250, 251
`\int_div_truncate:nn` 192, 411, 413, 414, 415, 424, 426, 441
`\int_eval:n` 101, 106, 155,
 184, 221, 280, 404, 480, 508, 515, 517, 519, 521, 532, 564
`\int_if_exist:cTF` 90, 276
`\int_incr:N` 478, 482
`\int_mod:nn` ... 177, 408, 421, 442, 463, 497, 525, 532, 705
`\int_new:N` 465, 836, 837
`\int_set:Nn` 824, 825, 827, 829, 830
`\int_set_eq:NN` 466
`\int_step_function:nnnN` 832
`\int_step_inline:nnnn` 555, 568, 708
`\int_to_arabic:n` 362, 448
`\int_zero:N` 471

K

keys commands:

`\l_keys_choice_tl` 858
`\keys_define:nn` 598, 852
`\keys_set:nn` 23, 84, 270, 291, 905, 908, 913, 914

M

mark commands:

`\q_mark` 40, 42

msg commands:

`\msg_error:nn` 11
`\msg_error:nnn` 864
`\msg_error:nnx` 766
`\msg_expandable_error:nnn` 95
`\msg_new:nnn` 3, 96
`\msg_new:nnnn` 789, 900

N

`\NewDocumentCommand` 20, 81, 267, 288, 469, 903

nil commands:

`\q_nil` 9, 28, 32, 40, 205

nine commands:

`\c_nine` 168

`null` 3, 21

O

one commands:

`\c_one` . 147, 161, 184, 221, 242, 333, 398, 411, 424, 487, 833
`\c_one_hundred` 414

or commands:

`\or:` 194, 195, 196, 379, 380, 381, 382, 383, 384

P

prg commands:

`\prg_do_nothing:` 810
`\ProcessKeysOptions` 909

prop commands:

`\prop_clear:N` 781
`\prop_get:NnNF` 644
`\prop_get:NnNT` 637
`\prop_get:NnNTF` 642, 661
`\prop_gset_eq:cN` 785
`\prop_if_exist:cTF` 751
`\prop_map_function:NN` 624, 625, 656
`\prop_map_inline:Nn` 721
`\prop_new:c` 782
`\prop_new:N` 618, 619, 620, 621
`\prop_put:Nnn` 601, 605, 608, 612, 615, 617
`\prop_put_if_new:Nnn` 604, 611
`\prop_set_eq:Nc` 754

Q

quark commands:

`\quark_if_nil:nTF` 31, 38, 44
`\quark_if_recursion_tail_stop:N` 210, 325
`\quark_if_recursion_tail_stop_do:Nn` 148

R

recursion commands:

`\q_recursion_stop` 119, 144, 150, 205, 306
`\q_recursion_tail` 9, 119, 205, 306
`\RequirePackage` 12
`reset` 4, 21

S

seven commands:

`\c_seven` 417, 429

six commands:

`\c_six` 414

stop commands:

`\q_stop` 28, 29, 32, 36, 40, 42, 350, 352, 354, 374, 375, 432, 435

str commands:

`\str_case:onTF` 822
`\str_fold_case:V` 858
`\str_if_empty:NT` 910
`\str_if_eq:NNF` 744
`\str_new:N` 786, 787, 899
`\str_set:Nx` 813, 840, 841, 857
`\str_set_eq:NN` 752, 814
`style` 3, 21

sys commands:

`\sys_if_engine luatex_p:` 798
`\sys_if_engine uptex_p:` 799
`\sys_if_engine xetex_p:` 797

T

ten commands:

\c_ten 403, 411, 424

T_EX and L^AT_EX₂_ε commands:

\ifpackagelater 10
 \zhdig 10, 280
 \zhnum 7, 112
 \pagenumbering 2, 2
 \tiangan 4
 \zhdate 2, 2
 \zhdig 1, 2, 4
 \zhdigits 1, 1, 1, 1, 3, 3, 4
 \zhdizhi 2
 \zhganzhi 2
 \zhganzhinian 3
 \zhnum 1, 2, 4
 \zhnumber 1, 1, 3, 3, 4
 \zhnumExtendScaleMap 3, 3
 \zhnumsetup 1, 3, 4
 \zhtiangan 2
 \zhtime 2
 \zhweekday 2

tex commands:

\tex_day:D 357
 \tex_ignorespaces:D 906
 \tex_month:D 357
 \tex_number:D 116, 192, 303
 \tex_time:D 441, 442
 \tex_year:D 357

three commands:

\c_three 250, 397

time 3

tl commands:

__tl_act:NNNnn 7
 \tl_const:cn 468
 \tl_const:cx 701, 710, 718, 722
 \tl_count:n 175
 \tl_if_blank:nF 51
 \tl_if_blank:nTF 64, 70
 \tl_if_exist:cF 481
 \tl_put_right:Nx 545, 557, 570, 575, 583, 592
 \tl_set:cn 483, 652, 703
 \tl_set:cx 646, 677, 711, 725
 \tl_set:Nn 540, 645
 \tl_set:Nx 479, 685, 687, 689, 691, 693, 695, 697
 \tl_set_eq:NN 740

tmpa commands:

\l_tmpa_int 471, 478, 480
 \l_tmpa_tl 479, 481, 483, 540,
 545, 557, 570, 575, 583, 592, 598, 637, 638, 642, 649, 661

tmpb commands:

\l_tmpb_tl 644, 645, 649

\TrimSpaces 469

true commands:

\c_true_bool 204, 219, 220, 802

twelve commands:

\c_twelve 398

two commands:

\c_two 180, 233, 249

U

use commands:

\use:c 456, 504, 507, 508, 525, 529
 \use:n 118, 126, 138, 150, 194, 305, 312, 329
 \use:x 433, 595
 \use_i:nn 169, 212, 803
 \use_i:nnn 196
 \use_i_ii:nnn 195
 \use_ii:nn 171, 807
 \use_none:n 253
 \use_none:nnn 767

Z

zero commands:

\c_zero ... 100, 103, 120, 158, 183, 211, 217, 226, 229,
 230, 237, 238, 243, 258, 340, 492, 503, 506, 524, 527, 707
 \zhcurrtime 2, 13, 438
 \zhdate 2, 11, 348
 \zhdig 2, 4, 10, 260
 \zhdigits 1, 4, 10, 281
 \zhdigitsoptions 10, 285, 288
 \zhdigwithoptions 264, 267
 \zhdizhi 2, 15, 516
 \zhganzhi 2, 15, 518
 \zhganzhinian 3, 15, 520
 \zhnum 2, 4, 6, 74

zhnum commands:

\l_zhnum_active_char_bool 819, 844, 849, 850, 890
 \l_zhnum_ancient_bool 233, 248, 868, 873, 878
 \c_zhnum_and_tl 54
 \l_zhnum_arabic_sep_tl 362, 448, 897
 \zhnum_assgin_const: 630, 654
 \zhnum_assgin_const_tl:cx 663, 671, 718, 725
 \zhnum_assgin_ganzhi:nn 625, 702
 \zhnum_blank_to_zero:n 6, 46, 48, 56, 58, 63, 68
 \l_zhnum_byte_max_int 827, 830, 834, 837
 \l_zhnum_byte_min_int 824, 825, 829, 833, 836
 \l_zhnum_cfg_map_finan_prop
 17, 608, 611, 620, 637, 661, 776
 \l_zhnum_cfg_map_ganzhi_prop
 17, 615, 617, 621, 625, 721, 777
 \l_zhnum_cfg_map_prop . 17, 601, 604, 618, 624, 656, 774
 \l_zhnum_cfg_map_var_prop
 17, 605, 612, 619, 642, 644, 775
 \l_zhnum_cfg_str
 743, 744, 751, 752, 774, 775, 776, 777, 786
 \zhnum_check_financial:nn 17, 656, 659
 \zhnum_check_simp:nn 17, 624, 634
 _zhnum_check_simp_aux:nn 636, 638, 640
 \zhnum_counter:n 6, 77, 85, 88
 _zhnum_counter_error:n 92, 94, 278
 _zhnum_date:www 350, 354

_zhnum_date_aux:nnn	12, 355, 358, 372	_zhnum_output_digits:NN	323, 336
_zhnum_date_aux:Nnnnn	362, 364	zhnum_parse_config:	17, 622, 746, 816
_zhnum_date_aux:NNnnnn	361, 365, 366	zhnum_parse_number:f	101, 106
_zhnum_date_aux:Vnn	357	zhnum_parse_number:n	8, 174, 188
\c_zhnum_day_tl	370	zhnum_parse_number:nn	8, 163, 175, 176
\l_zhnum_day_tl	698	_zhnum_parse_number:nnn	177, 178
zhnum_decimal:nn	6, 33, 61	\c_zhnum_parts_tl	47, 57
zhnum_digit_map:n	14, 181, 228, 234, 235, 253, 254, 258, 455	zhnum_process_number:NNNNNN	9, 215, 224
zhnum_digits:Nn	11, 284, 292, 296, 298, 300	_zhnum_prop_gset_eq:Nn	769, 784
zhnum_digits_counter:n	263, 271, 274	_zhnum_prop_initial:Nn	761, 779
zhnum_digits_null:f	280	_zhnum_read_abs_loop:Nw	7, 140, 144, 149
zhnum_digits_null:n	10, 297, 299, 361	_zhnum_read_digits:w	302, 331
zhnum_digits_null:v	277	_zhnum_read_digits_loop:NN	315, 320, 328
zhnum_digits_zero:n	10, 66, 295	_zhnum_read_integer:www	8, 115, 156
\c_zhnum_dot_tl	63, 326	_zhnum_read_sign_loop:N	117, 122, 125
\l_zhnum_encoding_str	813, 814, 822, 840, 843, 848, 857, 859, 899, 910	_zhnum_read_sign_loop:NN	304, 308, 311
_zhnum_fraction:www	5, 40, 42	_zhnum_read_zeros_loop:N	130, 134, 137
\c_zhnum_fri_tl	384	\l_zhnum_reset_bool	628, 631, 760, 788
\l_zhnum_fri_tl	693	zhnum_reset_config:	20, 816, 847, 889
zhnum_ganzhi:fnn	515, 517, 519	_zhnum_result:nn	120, 152, 153, 154
zhnum_ganzhi:nnn	15, 512, 513, 893, 895	\c_zhnum_sat_tl	378
zhnum_ganzhi_cyclic:nnn	15, 490, 893	\l_zhnum_sat_tl	695
_zhnum_ganzhi_cyclic_mod:fnnn	496	\l_zhnum_scale_int	463, 465, 466, 482
_zhnum_ganzhi_cyclic_mod:nnnn	15, 501, 511	zhnum_scale_map:n	14, 216, 457, 463, 464
zhnum_ganzhi_nian:f	521	zhnum_scale_map_hook:n	460, 467, 474
zhnum_ganzhi_nian:n	15, 522, 538	zhnum_scale_map_loop:n	14, 462, 467
zhnum_ganzhi_normal:nnn	14, 485, 512, 895	zhnum_set_active:	819, 820
\c_zhnum_hour_tl	452	zhnum_set_alias:	19, 657, 728
\c_zhnum_hundred_tl	235, 734	zhnum_set_alias:NN	727, 730, 732, 734, 736, 740
zhnum_if_digit:NTF	8, 146, 166, 322	zhnum_set_catcode:	20, 763, 810, 818
zhnum_if_unicode_engine:TF	20, 803, 807, 912	zhnum_set_cfg_name:Nn	20, 743, 811, 838
zhnum_if_unicode_engine_p:	20, 802, 806, 809	zhnum_set_digits_map:nn	17, 600, 923, 925, 927, 930, 932, 933, 934, 935, 936, 937, 938, 939, 940, 941, 942, 943, 944, 945, 946, 947, 953, 955, 956, 964, 965, 967, 968, 969, 970, 971, 979, 980, 982, 988, 990, 991, 992, 998, 1000, 1001
zhnum_input_cfg:n	753, 756	zhnum_set_digits_map:nnn	17, 602, 919, 920, 949, 950, 958, 959, 960, 961, 973, 974, 975, 976, 984, 985, 994, 995
zhnum_int:c	91	zhnum_set_dizhi_map:nn	17, 616, 1038, 1039, 1040, 1041, 1042, 1043, 1044, 1045, 1046, 1047, 1048, 1049
zhnum_int:n	6, 98, 111, 112, 361, 447	zhnum_set_financial_map:nn	17, 607, 1002, 1003, 1004, 1005, 1011, 1013, 1014, 1020, 1022, 1023, 1024, 1025, 1026, 1027
zhnum_integer:n	7, 11, 39, 113	zhnum_set_financial_map:nnn	17, 609, 1007, 1008, 1016, 1017
_zhnum_integer_or_fraction:www	5, 32, 36	zhnum_set_scale:n	14, 472, 476
\l_zhnum_last_cfg_str	744, 752, 787	zhnum_set_tiangang_map:nn	17, 614, 1028, 1029, 1030, 1031, 1032, 1033, 1034, 1035, 1036, 1037
zhnum_load_cfg:n	19, 741, 748	zhnum_set_week_day:	17, 627, 683
zhnum_load_cfg:o	848, 859	zhnum_set_zero:	17, 626, 675
_zhnum_loop_end:wnn	148, 154	\l_zhnum_simp_bool	648, 881, 882
\c_zhnum_minus_tl	105, 162, 334	zhnum_split_number:fn	184
\l_zhnum_minus_tl	542	zhnum_split_number:nn	9, 201, 202, 207
\c_zhnum_minute_tl	453	zhnum_split_number:NNfNNNNw	219, 220
\c_zhnum_mon_tl	380	zhnum_split_number:NNnNNNNw	9, 204, 208, 223
\l_zhnum_mon_tl	685	_zhnum_split_number_aux:nnn	8, 185, 189
\c_zhnum_month_tl	369		
\l_zhnum_normal_bool	665, 869, 874, 879		
\l_zhnum_null_bool	679, 884		
\l_zhnum_null_tl	543, 680		
zhnum_number:f	16, 24		
zhnum_number:n	5, 27, 35, 53, 72		
_zhnum_number:www	5, 28, 29		
_zhnum_output:nnwnn	147, 152		

_zhnum_split_number_aux:wnn	191, 200	\c_zhnum_weekday_tl	686, 688, 690, 692, 694, 696, 698
\c_zhnum_sun_tl	379	\c_zhnum_year_tl	368
\l_zhnum_sun_tl	697	\zhnum_Zeller:nnn	12, 377, 387
\c_zhnum_ten_tl	256, 732	_zhnum_Zeller_aux:Nnnn	391, 392, 395
\c_zhnum_thousand_tl	228, 736	\zhnum_Zeller_aux:Nnnn	12
\c_zhnum_thu_tl	383	\zhnum_Zeller_Gregorian:nnn	13, 391, 406
\l_zhnum_thu_tl	691	\zhnum_Zeller_Julian:nnn	13, 392, 419
_zhnum_time:ww	432, 435	\zhnum_zero_mod:nn	704, 713, 714
_zhnum_time_aux:nn	14, 437, 440, 444	_zhnum_zero_mod_aux:nn	705, 706
_zhnum_time_aux:Nnnn	14, 447, 448, 450	\c_zhnum_zero_tl	65, 71, 108, 159, 214, 227, 230, 238, 730
\l_zhnum_time_bool	360, 446, 886, 887	\zhnumber	1, 4, 5, 13
\c_zhnum_tue_tl	381	\zhnumberwithoptions	5, 6, 17, 20
\l_zhnum_tue_tl	687	\zhnumExtendScaleMap	3, 14, 469
\zhnum_two_digits:n	12, 390, 401	\zhnumsetup	3, 22, 903
\zhnum_update_cfg:n	745, 749	\zhnumwithoptions	78, 81
_zhnum_update_cfg_prop:N	754, 761, 769, 772	\zhtiangan	2, 15, 514
\c_zhnum_wed_tl	382	\zhptime	2, 13, 431
\l_zhnum_wed_tl	689	\zhtoday	2, 11, 356
_zhnum_week_day:www	12, 352, 374, 375	\zhweekday	2, 12, 373