

zhnumber 宏包

李清

sobenlee@gmail.com

2017/08/07 v2.5*

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 GBK|Big5|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} \\
2 \zhnumber{2 012 020 120} \\
3 \zhnumber{2,012,020,120} \\
4 \zhnumber{2012.020120} \\
5 \zhnumber{2012.} \\
6 \zhnumber{.2012} \\
7 \zhnumber{20120/20120} \\
8 \zhnumber{/2012} \\
9 \zhnumber{2012/} \\
10 \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} \\
2 \zhdigits*{2012020120}
```

*ctex-kit rev. 858feab.

<hr/> <code>\zhnum</code> ☆ <hr/>	<code>\zhnum {⟨counter⟩}</code> <code>\pagenumbering {zhnum}</code> 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
Updated: 2016-05-01	二 <div>1<div><code>\zhnum{section}</code></div></div>
<hr/> <code>\zhdig</code> ☆ <hr/>	<code>\zhdig {⟨counter⟩}</code> <code>\pagenumbering {zhdig}</code> 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
New: 2016-05-01	二 <div>1<div><code>\zhdig{section}</code></div></div>
<hr/> <code>\zhweekday</code> ☆ <hr/>	<code>\zhweekday {⟨yyyy/mm/dd⟩}</code> 输出日期当天的星期。例如
New: 2012-05-25	星期日 <div>1<div><code>\zhweekday{2012/5/20}</code></div></div>
<hr/> <code>\zhdate</code> ☆ <hr/>	<code>\zhdate {⟨yyyy/mm/dd⟩}</code> <code>\zhdate * {⟨yyyy/mm/dd⟩}</code> 以中文格式输出日期,其中带 * 的命令还输出星期。例如
New: 2012-05-25	2012 年 5 月 21 日 2012 年 5 月 21 日星期一 <div>1<div><code>\zhdate{2012/5/21}\</code></div><div>2<div><code>\zhdate*{2012/5/21}</code></div></div></div>
<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	2017 年 8 月 7 日 <div>1<div><code>\zhtoday</code></div></div>
<hr/> <code>\zhtime</code> ☆ <hr/>	<code>\zhtime {⟨hh:mm⟩}</code> 以中文格式输出时间。例如
New: 2012-05-25	23 时 56 分 <div>1<div><code>\zhtime{23:56}</code></div></div>
<hr/> <code>\zhcurrtime</code> ☆ <hr/>	输出当前的时间。例如
New: 2012-05-25	17 时 11 分 <div>1<div><code>\zhcurrtime</code></div></div>
<hr/> <code>\zhtiangan</code> ☆ <hr/>	<code>\zhtiangan {⟨number⟩}</code> 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
New: 2015-05-20	甲 乙 丙 丁 戊 癸 <div>1<div><code>\zhtiangan{1} \zhtiangan{2} \zhtiangan{3}</code></div><div>2<div><code>\zhtiangan{4} \zhtiangan{5} \zhtiangan{10}</code></div></div></div>
<hr/> <code>\zhdizhi</code> ☆ <hr/>	<code>\zhdizhi {⟨number⟩}</code> 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
New: 2015-05-20	子 丑 寅 卯 辰 亥 <div>1<div><code>\zhdizhi{1} \zhdizhi{2} \zhdizhi{3}</code></div><div>2<div><code>\zhdizhi{4} \zhdizhi{5} \zhdizhi{12}</code></div></div></div>
<hr/> <code>\zhgan zhi</code> ☆ <hr/>	<code>\zhgan zhi {⟨number⟩}</code> 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
New: 2015-05-20	甲子 乙丑 丙寅 丁卯 戊辰 癸亥 <div>1<div><code>\zhgan zhi{1} \zhgan zhi{2} \zhgan zhi{3} \</code></div><div>2<div><code>\zhgan zhi{4} \zhgan zhi{5} \zhgan zhi{60}</code></div></div></div>

<code>\zhganzhinian</code> ☆	<code>\zhganzhinian {⟨year⟩}</code>
New: 2015-05-20	输出公元纪年 $\langle year \rangle$ 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 丁酉
	<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> 。 $\langle character_i \rangle$ 设置 $10^{4(i+11)}$ 。若给出可选项 $\langle character \rangle$, 则当数字大于 $10^{4(n+12)} - 1$ 时, 统一用 $\langle character \rangle$ 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 $\langle key \rangle$ 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, $\langle Arabic \rangle$ 为阿拉伯数字, 而 $\langle Chinese \rangle$ 为中文数字。例如 二〇一七年八月七日十七时十一分
	<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, 以𠫪输出 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-cyclic`

New: 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 \bigcirc , `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。

`reset`

Updated: 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)。

`activechar`

New: 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 } { l3-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  \ifpackageafter { expl3 } { 2017/07/19 } { }
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 }

```

`\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 }

```

`\zhnum_number:n`
`__zhnum_number:www`

先判断输入的是小数还是分数。

```

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 }

```

`__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 }

```

`__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 }

```

`\zhnum_decimal:n` 对小数进行预处理。

```

61 \cs_new:Npn \zhnum_decimal:n #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 }

```

`\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 }

```

`\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 }

```

`\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 }

```

`\@zhnum` 用于支持 `\pagenumbering{zhnum}`。

```

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

```

`\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 }

```

`__zhnum_read_abs_loop:Nw`

当数字很大时, `l3bigint` 的实现会造成 $\text{T}_{\text{E}}\text{X}$ 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 }

```

`__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 }

```

\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 }

```

```

\zhnum_parse_number:n 174 \cs_new:Npn \zhnum_parse_number:n #1
\zhnum_parse_number:nn 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 }

```

__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 } }

```

\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:NNnNNNw \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 }

```

\zhnum_split_number:NNnNNNw 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:NNnNNNw #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:NNfNNNw \c_false_bool \c_true_bool }
220   { \zhnum_split_number:NNfNNNw \c_true_bool \c_false_bool }
221   { \int_eval:n { #3 - \c_one } }
222 }
223 \cs_generate_variant:Nn \zhnum_split_number:NNnNNNw { NNf }

```


`\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_lazy_and:nnTF
233         { \l__zhnum_ancient_bool }
234         { \int_compare_p:nNn {#2} = \c_two }
235         { \zhnum_digit_map:n { #2 00 } }
236         { \zhnum_digit_map:n {#2} \c__zhnum_hundred_tl }
237     }
238   \int_compare:nNnTF {#3} = \c_zero
239     { \int_compare:nNnF { #2 * #4 } = \c_zero { \c__zhnum_zero_tl } }
240     {
241       \bool_lazy_all:nF
242       {
243         { \int_compare_p:nNn {#3} = \c_one }
244         { \int_compare_p:nNn {#1#2} = \c_zero }
245         {#6}
246         {#5}
247       }
248       {
249         \bool_lazy_and:nnTF
250         { \l__zhnum_ancient_bool }
251         { \int_compare_p:n { \c_one < #3 < \c_five } }
252         { \zhnum_digit_map:n { #3 0 } \use_none:n }
253         { \zhnum_digit_map:n {#3} }
254       }
255       \c__zhnum_ten_tl
256     }
257   \int_compare:nNnF {#4} = \c_zero { \zhnum_digit_map:n {#4} }
258 }

```

`\zhdig` 用于将 L^AT_EX 计数器按中文数字串输出。

```

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

```

`\@zhdig` 用于支持 `\pagenumbering{zhdig}`。

```

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

```

`\zhdigits` 将输入的数字输出为中文数字串输出。`\zhdigitwithoptions`

```

280 \DeclareExpandableDocumentCommand \zhdigits { +s +o +m }
281 {
282   \IfNoValueTF {#2}

```

```

283     { \zhnum_digits:Nn #1 }
284     { \zhdigitsoptions {#1} {#2} }
285     {#3}
286   }
287 \NewDocumentCommand \zhdigitsoptions { +m +m +m }
288 {
289   \group_begin:
290   \keys_set:nn { zhnum / options } {#2}
291   \zhnum_digits:Nn #1 {#3}
292   \group_end:
293 }

```

\zhnum_digits_zero:n
\zhnum_digits_null:n

快捷方式。

```

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

```

\zhnum_digits:Nn

与 \zhnum_integer:n 类似,但不用去掉多余的零。

```

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

```

```

345 \cs_end:
346 }

```

\zhdate 输出中文日期。

```

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

```

\zhtoday 输出当天日期。

```

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

```

```

\__zhnum_date_aux:nnn 357 \cs_new_nopar:Npn \__zhnum_date_aux:nnn
358 {
359   \bool_if:NTF \l__zhnum_time_bool
360     { \__zhnum_date_aux:NNnnnn \zhnum_digits_null:n \zhnum_int:n { } }
361     { \__zhnum_date_aux:Nnnnn \int_to_arabic:n { \l__zhnum_arabic_sep_tl } }
362 }
363 \cs_new:Npn \__zhnum_date_aux:Nnnnn #1
364 { \__zhnum_date_aux:NNnnnn #1#1 }
365 \cs_new:Npn \__zhnum_date_aux:NNnnnn #1#2#3#4#5#6
366 {
367   #1 {#4} #3 \c__zhnum_year_tl #3
368   #2 {#5} #3 \c__zhnum_month_tl #3
369   #2 {#6} #3 \c__zhnum_day_tl
370 }
371 \cs_generate_variant:Nn \__zhnum_date_aux:nnn { V }

```

\zhweekday 输出星期

```

372 \cs_new:Npn \zhweekday #1
373 { \__zhnum_week_day:www #1 \q_stop }

```

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

```

374 \cs_new:Npn \__zhnum_week_day:www #1/#2/#3 \q_stop
375 {
376   \if_case:w \zhnum_Zeller:nnn {#1} {#2} {#3} \exp_stop_f:
377     \c__zhnum_sat_tl
378     \or: \c__zhnum_sun_tl
379     \or: \c__zhnum_mon_tl
380     \or: \c__zhnum_tue_tl
381     \or: \c__zhnum_wed_tl
382     \or: \c__zhnum_thu_tl
383     \or: \c__zhnum_fri_tl
384   \fi:
385 }

```

\zhnum_Zeller:nnn 用 Zeller 公式¹ 计算星期几。

```

\zhnum_Zeller_aux:Nnnn 386 \cs_new:Npn \zhnum_Zeller:nnn #1#2#3
\zhnum_two_digits:n 387 {
388   \int_compare:nNnTF
389     { #1 \zhnum_two_digits:n {#2} \zhnum_two_digits:n {#3} } > { 1582 10 04 }
390     { \__zhnum_Zeller_aux:Nnnn \zhnum_Zeller_Gregorian:nnn }
391     { \__zhnum_Zeller_aux:Nnnn \zhnum_Zeller_Julian:nnn }
392     {#1} {#2} {#3}
393 }
394 \cs_new:Npn \__zhnum_Zeller_aux:Nnnn #1#2#3#4
395 {
396   \int_compare:nNnTF {#3} < \c_three

```

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

```

397     { #1 { #2 - \c_one } { #3 + \c_twelve } {#4} }
398     { #1 {#2} {#3} {#4} }
399   }
400 \cs_new:Npn \zhnum_two_digits:n #1
401 {
402   \int_compare:nNtT {#1} < \c_ten { 0 }
403   \int_eval:n {#1}
404 }

```

\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 $Y --$ 。

```

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

```

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

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

```

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

```

\zhptime 输出时间。

```

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

```

\zhcurtime 输出当前时间。

```

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

```

```

__zhnum_time_aux:nn 443 \cs_new_nopar:Npn \__zhnum_time_aux:nn
__zhnum_time_aux:Nnnn 444 {
445   \bool_if:NTF \l__zhnum_time_bool
446     { \__zhnum_time_aux:Nnnn \zhnum_int:n { } }
447     { \__zhnum_time_aux:Nnnn \int_to_arabic:n { \l__zhnum_arabic_sep_tl } }
448   }
449 \cs_new:Npn \__zhnum_time_aux:Nnnn #1#2#3#4
450 {
451   #1 {#3} #2 \c__zhnum_hour_tl #2
452   #1 {#4} #2 \c__zhnum_minute_tl
453 }

```

\zhnum_digit_map:n 阿拉伯数字与中文数字的映射。

```

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

```

\zhnum_scale_map:n 大数系统的映射。

```

\zhnum_scale_map_loop:n 456 \cs_new:Npn \zhnum_scale_map:n #1
457 {
458   \cs_if_exist_use:cF { c__zhnum_s #1 _tl }
459   { \zhnum_scale_map_hook:n {#1} }
460 }
461 \cs_new:Npn \zhnum_scale_map_loop:n #1
462 { \zhnum_scale_map:n { \int_mod:nn {#1} \l__zhnum_scale_int } }
463 \cs_generate_variant:Nn \zhnum_scale_map:n { f }
464 \int_new:N \l__zhnum_scale_int
465 \int_set_eq:NN \l__zhnum_scale_int \c_eleven
466 \cs_new_eq:NN \zhnum_scale_map_hook:n \zhnum_scale_map_loop:n
467 \tl_const:cn { c__zhnum_s0_tl } { }

```

\zhnumExtendScaleMap 扩展进位系统。

```

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

```

```

\zhnum_set_scale:n 475 \cs_new_protected:Npn \zhnum_set_scale:n #1
476 {
477   \int_incr:N \l_tmpa_int
478   \tl_set:Nx \l_tmpa_tl
479   { c__zhnum_s \int_eval:n { \l_tmpa_int + \c_eleven } _tl }
480   \tl_if_exist:cF { \l_tmpa_tl }
481   { \int_incr:N \l__zhnum_scale_int }
482   \tl_set:cn { \l_tmpa_tl } {#1}
483 }

```

\zhnum_ganzhi_normal:nnn 保证干支的参数为正数。

```

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

```

\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 489 \cs_new:Npn \zhnum_ganzhi_cyclic:nnn #1#2#3
490 {
491   \int_compare:nNnF {#1} = \c_zero
492   {
493     \cs_if_exist_use:cF { c__zhnum_ #2 _ #1 _tl }
494     {
495       \__zhnum_ganzhi_cyclic_mod:fnnn
496       { \int_mod:nn {#1} {#3} } {#1} {#2} {#3}

```

```

497     }
498   }
499 }
500 \cs_new:Npn \__zhnum_ganzhi_cyclic_mod:nnnn #1#2#3#4
501 {
502   \int_compare:nNnTF {#2} > \c_zero
503   { \use:c { c__zhnum_ #3 _ #1 _tl } }
504   {
505     \int_compare:nNnTF {#1} = \c_zero
506     { \use:c { c__zhnum_ #3 _ 1 _tl } }
507     { \use:c { c__zhnum_ #3 _ \int_eval:n { #1 + #4 + 1 } _tl } }
508   }
509 }
510 \cs_generate_variant:Nn \__zhnum_ganzhi_cyclic_mod:nnnn { f }

```

`\zhnum_ganzhi:nnn` 默认不对超出范围的数字取模。

```

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

```

`\zhtiangan` 天干。

```

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

```

`\zhdizhi` 地支。

```

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

```

`\zhganzhi` 干支。

```

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

```

`\zhganzhinian` 干支纪年。

```

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

```

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

```

521 \cs_new:Npn \zhnum_ganzhi_nian:n #1
522 {
523   \int_compare:nNnTF {#1} > \c_zero
524   { \use:c { c__zhnum_ganzhi_ \int_mod:nn { #1 + 57 } { 60 } _tl } }
525   {
526     \int_compare:nNnF {#1} = \c_zero
527     {
528       \use:c
529       {
530         c__zhnum_ganzhi_
531         \int_eval:n { \int_mod:nn { #1 - 2 } { 60 } + 60 }
532         _tl
533       }
534     }
535   }
536 }
537 \cs_generate_variant:Nn \zhnum_ganzhi_nian:n { f }

```

根据需要设置中文阿拉伯数字。

```

538 \group_begin:
539 \tl_set:Nn \l_tmpa_tl
540 {
541   - .tl_set:N = \l_zhnum_minus_tl ,
542   -0 .tl_set:N = \l_zhnum_null_tl ,
543 }
544 \tl_put_right:Nx \l_tmpa_tl
545 {
546   E2 .tl_set:N = \exp_not:c { l_zhnum_ 100 _tl } ,

```

```

547     E3 .tl_set:N = \exp_not:c { l__zhnum_ 1000 _tl } ,
548     FE2 .tl_set:N = \exp_not:c { l__zhnum_financial_ 100 _tl } ,
549     FE3 .tl_set:N = \exp_not:c { l__zhnum_financial_ 1000 _tl } ,
550     D11 .tl_set:N = \exp_not:c { l__zhnum_dizhi_ 11 _tl } ,
551     D12 .tl_set:N = \exp_not:c { l__zhnum_dizhi_ 12 _tl } ,
552     E44 .tl_set:N = \exp_not:c { l__zhnum_ s11 _tl } ,
553   }
554   \int_step_inline:nnnn { 1 } { 1 } { 10 }
555   {
556     \tl_put_right:Nx \l_tmpa_tl
557     {
558       #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } ,
559       F#1 .tl_set:N = \exp_not:c { l__zhnum_financial_ #1 _tl } ,
560       T#1 .tl_set:N = \exp_not:c { l__zhnum_tiangang_ #1 _tl } ,
561       D#1 .tl_set:N = \exp_not:c { l__zhnum_dizhi_ #1 _tl } ,
562       GZ#1 .tl_set:N = \exp_not:c { l__zhnum_ganzhi_ #1 _tl } ,
563       E \int_eval:n { #1 * 4 }
564       .tl_set:N = \exp_not:c { l__zhnum_ s#1 _tl } ,
565     }
566   }
567   \int_step_inline:nnnn { 11 } { 1 } { 60 }
568   {
569     \tl_put_right:Nx \l_tmpa_tl
570     { GZ#1 .tl_set:N = \exp_not:c { l__zhnum_ganzhi_ #1 _tl } , }
571   }
572   \clist_map_inline:nn { 0 , 100 , 1000 }
573   {
574     \tl_put_right:Nx \l_tmpa_tl
575     {
576       #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } ,
577       F#1 .tl_set:N = \exp_not:c { l__zhnum_financial_ #1 _tl } ,
578     }
579   }
580   \clist_map_inline:nn { 20 , 30 , 40 , 200 }
581   {
582     \tl_put_right:Nx \l_tmpa_tl
583     { #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } , }
584   }
585   \clist_map_inline:nn
586   {
587     dot , and , parts , year , month , day , weekday , hour , minute
588     mon , tue , wed , thu , fri , sat , sun
589   }
590   {
591     \tl_put_right:Nx \l_tmpa_tl
592     { #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } , }
593   }
594   \use:x
595   {
596     \group_end:
597     \keys_define:nn { zhnum / options } { \exp_not:o \l_tmpa_tl }
598   }

```

将配置文件中的中文数字保存到 prop 变量中。

```

\zhnum_set_digits_map:nn
\zhnum_set_digits_map:nnn
\zhnum_set_financial_map:nn
\zhnum_set_financial_map:nnn
\zhnum_set_tiangang_map:nn
  \zhnum_set_dizhi_map:nn
  \l__zhnum_cfg_map_prop
\l__zhnum_cfg_map_var_prop
\l__zhnum_cfg_map_finan_prop
\l__zhnum_cfg_map_ganzhi_prop
599 \cs_new_protected:Npn \zhnum_set_digits_map:nn #1#2
600 { \prop_put:Nnn \l__zhnum_cfg_map_prop {#1} {#2} }
601 \cs_new_protected:Npn \zhnum_set_digits_map:nnn #1#2#3
602 {
603   \prop_put_if_new:Nnn \l__zhnum_cfg_map_prop {#1} {#3}
604   \prop_put:Nnn \l__zhnum_cfg_map_var_prop {#1#2} {#3}
605 }
606 \cs_new_protected:Npn \zhnum_set_financial_map:nn #1#2
607 { \prop_put:Nnn \l__zhnum_cfg_map_finan_prop {#1} {#2} }
608 \cs_new_protected:Npn \zhnum_set_financial_map:nnn #1#2#3
609 {
610   \prop_put_if_new:Nnn \l__zhnum_cfg_map_finan_prop {#1} {#3}
611   \prop_put:Nnn \l__zhnum_cfg_map_var_prop { financial_#1#2 } {#3}

```

```

612 }
613 \cs_new_protected:Npn \zhnum_set_tiangang_map:nn #1#2
614 { \prop_put:Nnn \l__zhnum_cfg_map_ganzhi_prop { tiangan_#1 } {#2} }
615 \cs_new_protected:Npn \zhnum_set_dizhi_map:nn #1#2
616 { \prop_put:Nnn \l__zhnum_cfg_map_ganzhi_prop { dizhi_#1 } {#2} }
617 \prop_new:N \l__zhnum_cfg_map_prop
618 \prop_new:N \l__zhnum_cfg_map_var_prop
619 \prop_new:N \l__zhnum_cfg_map_finan_prop
620 \prop_new:N \l__zhnum_cfg_map_ganzhi_prop

```

将 prop 表转化到单独的 tl 变量。

```

\zhnum_parse_config:
\zhnum_check_simp:nn
\zhnum_check_financial:nn
\zhnum_set_zero:
\zhnum_set_week_day:
621 \cs_new_protected_nopar:Npn \zhnum_parse_config:
622 {
623   \prop_map_function:NN \l__zhnum_cfg_map_prop \zhnum_check_simp:nn
624   \prop_map_function:NN \l__zhnum_cfg_map_ganzhi_prop \zhnum_assgin_ganzhi:nn
625   \zhnum_set_zero:
626   \zhnum_set_week_day:
627   \bool_if:NF \l__zhnum_reset_bool
628   {
629     \zhnum_assgin_const:
630     \bool_set_true:N \l__zhnum_reset_bool
631   }
632 }
633 \cs_new_protected:Npn \zhnum_check_simp:nn #1#2
634 {
635   \__zhnum_check_simp_aux:nn {#2} {#1}
636   \prop_get:NnNT \l__zhnum_cfg_map_finan_prop {#1} \l_tmpa_tl
637   { \exp_args:No \__zhnum_check_simp_aux:nn { \l_tmpa_tl } { financial_ #1 } }
638 }
639 \cs_new_protected:Npn \__zhnum_check_simp_aux:nn #1#2
640 {
641   \prop_get:NnNTF \l__zhnum_cfg_map_var_prop { #2 _trad } \l_tmpa_tl
642   {
643     \prop_get:NnNF \l__zhnum_cfg_map_var_prop { #2 _simp } \l_tmpb_tl
644     { \tl_set:Nn \l_tmpb_tl {#1} }
645     \tl_set:cx { l__zhnum_ #2 _tl }
646     {
647       \exp_not:n { \bool_if:NTF \l__zhnum_simp_bool }
648       { \exp_not:o \l_tmpb_tl } { \exp_not:o \l_tmpa_tl }
649     }
650   }
651   { \tl_set:cn { l__zhnum_ #2 _tl } {#1} }
652 }
653 \cs_new_protected_nopar:Npn \zhnum_assgin_const:
654 {
655   \prop_map_function:NN \l__zhnum_cfg_map_prop \zhnum_check_financial:nn
656   \zhnum_set_alias:
657 }
658 \cs_new_protected:Npn \zhnum_check_financial:nn #1#2
659 {
660   \prop_get:NnNTF \l__zhnum_cfg_map_finan_prop {#1} \l_tmpa_tl
661   {
662     \zhnum_assgin_const_tl:cx { c__zhnum_ #1 _tl }
663     {
664       \exp_not:n { \bool_if:NTF \l__zhnum_normal_bool }
665       { \exp_not:c { l__zhnum_ #1 _tl } }
666       { \exp_not:c { l__zhnum_financial_ #1 _tl } }
667     }
668   }
669   {
670     \zhnum_assgin_const_tl:cx
671     { c__zhnum_ #1 _tl } { \exp_not:c { l__zhnum_ #1 _tl } }
672   }
673 }
674 \cs_new_protected_nopar:Npn \zhnum_set_zero:
675 {
676   \tl_set:cx { l__zhnum_0_tl }

```



```

677     {
678         \exp_not:n { \bool_if:NTF \l__zhnum_null_bool }
679         { \exp_not:o \l__zhnum_null_tl } { \exp_not:v { l__zhnum_0_tl } }
680     }
681 }
682 \cs_new_protected_nopar:Npn \zhnum_set_week_day:
683 {
684     \tl_set:Nx \l__zhnum_mon_tl
685     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_1_tl } }
686     \tl_set:Nx \l__zhnum_tue_tl
687     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_2_tl } }
688     \tl_set:Nx \l__zhnum_wed_tl
689     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_3_tl } }
690     \tl_set:Nx \l__zhnum_thu_tl
691     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_4_tl } }
692     \tl_set:Nx \l__zhnum_fri_tl
693     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_5_tl } }
694     \tl_set:Nx \l__zhnum_sat_tl
695     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_6_tl } }
696     \tl_set:Nx \l__zhnum_sun_tl
697     { \exp_not:N \c__zhnum_weekday_tl \exp_not:o \l__zhnum_day_tl }
698 }
699 \clist_map_inline:nn { mon , tue , wed , thu , fri , sat , sun }
700 { \tl_const:cx { c__zhnum_ #1 _tl } { \exp_not:c { l__zhnum_ #1 _tl } } }
701 \cs_new_protected:Npn \zhnum_assgin_ganzhi:nn #1#2
702 { \tl_set:cn { l__zhnum_ #1 _tl } {#2} }
703 \cs_new:Npn \zhnum_zero_mod:nn #1#2
704 { \exp_args:Nf \__zhnum_zero_mod_aux:nn { \int_mod:nn {#1} {#2} } {#2} }
705 \cs_new:Npn \__zhnum_zero_mod_aux:nn #1#2
706 { \int_compare:nNnTF {#1} = \c_zero {#2} {#1} }
707 \int_step_inline:nnnn { 1 } { 1 } { 60 }
708 {
709     \tl_const:cx { c__zhnum_ganzhi_ #1 _tl } { \exp_not:c { l__zhnum_ganzhi_ #1 _tl } }
710     \tl_set:cx { l__zhnum_ganzhi_ #1 _tl }
711     {
712         \exp_not:c { l__zhnum_tiangang_ \zhnum_zero_mod:nn {#1} { 10 } _tl }
713         \exp_not:c { l__zhnum_dizhi_ \zhnum_zero_mod:nn {#1} { 12 } _tl }
714     }
715 }
716 \cs_new_eq:cc { c__zhnum_ganzhi_ 0 _tl } { c__zhnum_ganzhi_ 60 _tl }
717 \cs_new_eq:NN \zhnum_assgin_const_tl:cx \tl_const:cx
718 \AtEndOfPackage
719 {
720     \prop_map_inline:Nn \l__zhnum_cfg_map_ganzhi_prop
721     { \tl_const:cx { c__zhnum_ #1 _tl } { \exp_not:c { l__zhnum_ #1 _tl } } }
722     \cs_new_eq:cc { c__zhnum_tiangang_ 0 _tl } { c__zhnum_tiangang_ 10 _tl }
723     \cs_new_eq:cc { c__zhnum_dizhi_ 0 _tl } { c__zhnum_dizhi_ 12 _tl }
724     \cs_set_eq:NN \zhnum_assgin_const_tl:cx \tl_set:cx
725 }

```

\zhnum_set_alias: 一些易于使用的别名。

```

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

```

\zhnum_load_cfg:n 根据选定编码载入配置文件。

```

740 \cs_new_protected:Npn \zhnum_load_cfg:n #1
741 {
742   \zhnum_set_cfg_name:Nn \l__zhnum_cfg_str {#1}
743   \str_if_eq:NNF \l__zhnum_cfg_str \l__zhnum_last_cfg_str
744   { \zhnum_update_cfg:n {#1} }
745   \zhnum_parse_config:
746 }
747 \cs_generate_variant:Nn \zhnum_load_cfg:n { o }
748 \cs_new_protected:Npn \zhnum_update_cfg:n #1
749 {
750   \prop_if_exist:cTF { g__zhnum_cfg_ \l__zhnum_cfg_str _prop }
751   { \str_set_eq:NN \l__zhnum_last_cfg_str \l__zhnum_cfg_str }
752   { \zhnum_input_cfg:n {#1} }
753   \__zhnum_update_cfg_prop:N \prop_set_eq:Nc
754 }
755 \cs_new_protected:Npn \zhnum_input_cfg:n #1
756 {
757   \file_if_exist:nTF { zhnumber - #1 .cfg }
758   {
759     \bool_set_false:N \l__zhnum_reset_bool
760     \__zhnum_update_cfg_prop:N \__zhnum_prop_initial:Nn
761     \group_begin:
762       \zhnum_set_catcode:
763       \file_input:n { zhnumber - #1 .cfg }
764       \__zhnum_update_cfg_prop:N \__zhnum_prop_gset_eq:Nn
765     \group_end:
766   }
767   { \msg_error:nxx { zhnumber } { file-not-found } {#1} }
768 }
769 \cs_new_protected:Npn \__zhnum_update_cfg_prop:N #1
770 {
771   #1 \l__zhnum_cfg_map_prop { g__zhnum_cfg_ \l__zhnum_cfg_str _prop }
772   #1 \l__zhnum_cfg_map_var_prop { g__zhnum_cfg_var_ \l__zhnum_cfg_str _prop }
773   #1 \l__zhnum_cfg_map_finan_prop { g__zhnum_cfg_finan_ \l__zhnum_cfg_str _prop }
774   #1 \l__zhnum_cfg_map_ganzhi_prop { g__zhnum_cfg_ganzhi_ \l__zhnum_cfg_str _prop }
775 }
776 \cs_new_protected:Npn \__zhnum_prop_initial:Nn #1#2
777 {
778   \prop_clear:N #1
779   \prop_new:c {#2}
780 }
781 \cs_new_protected:Npn \__zhnum_prop_gset_eq:Nn #1#2
782 { \prop_gset_eq:cN {#2} #1 }
783 \str_new:N \l__zhnum_cfg_str
784 \str_new:N \l__zhnum_last_cfg_str
785 \bool_new:N \l__zhnum_reset_bool
786 \msg_new:nnnn { zhnumber } { file-not-found }
787 { File~`#1'~not~found. }
788 {
789   The~requested~file~could~not~be~found~in~the~current~directory,~
790   in~the~TeX~search~path~or~in~the~LaTeX~search~path.
791 }

```

\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

```

792 \bool_lazy_any:nTF
793 {
794   { \sys_if_engine_xetex_p: }
795   { \sys_if_engine luatex_p: }
796   { \sys_if_engine_uptex_p: }
797 }
798 {
799   \cs_new_eq:NN \zhnum_if_unicode_engine_p: \c_true_bool
800   \cs_new_eq:NN \zhnum_if_unicode_engine:TF \use_i:nn
801 }

```

```

802 {
803   \cs_new_eq:NN \zhnum_if_unicode_engine_p: \c_false_bool
804   \cs_new_eq:NN \zhnum_if_unicode_engine:TF \use_i:nn
805 }

```

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

```

\zhnum_set_cfg_name:Nn
\zhnum_reset_config:
806 \if_predicate:w \zhnum_if_unicode_engine_p:
807   \cs_new_eq:NN \zhnum_set_catcode: \prg_do_nothing:
808   \cs_new_protected:Npn \zhnum_set_cfg_name:Nn #1#2
809   {
810     \str_set:Nx \l__zhnum_encoding_str {#2}
811     \str_set_eq:NN #1 \l__zhnum_encoding_str
812   }
813   \cs_new_eq:NN \zhnum_reset_config: \zhnum_parse_config:
814 \else:
815   \cs_new_protected_nopar:Npn \zhnum_set_catcode:
816   { \bool_if:NT \l__zhnum_active_char_bool { \zhnum_set_active: } }
817   \cs_new_protected_nopar:Npn \zhnum_set_active:
818   {
819     \str_case:onTF { \l__zhnum_encoding_str }
820     {
821       { gbk } { \int_set:Nn \l__zhnum_byte_min_int { "81 } }
822       { big5 } { \int_set:Nn \l__zhnum_byte_min_int { "A1 } }
823     }
824     { \int_set:Nn \l__zhnum_byte_max_int { "FE } }
825     {
826       \int_set:Nn \l__zhnum_byte_min_int { "E0 }
827       \int_set:Nn \l__zhnum_byte_max_int { "EF }
828     }
829     \int_step_function:nnnN
830     { \l__zhnum_byte_min_int } { \c_one }
831     { \l__zhnum_byte_max_int } \char_set_catcode_active:n
832   }
833   \int_new:N \l__zhnum_byte_min_int
834   \int_new:N \l__zhnum_byte_max_int
835   \cs_new_protected:Npn \zhnum_set_cfg_name:Nn #1#2
836   {
837     \str_set:Nx \l__zhnum_encoding_str {#2}
838     \str_set:Nx #1
839     {
840       \l__zhnum_encoding_str
841       \bool_if:NT \l__zhnum_active_char_bool { _active }
842     }
843   }
844   \cs_new_protected_nopar:Npn \zhnum_reset_config:
845   { \zhnum_load_cfg:o { \l__zhnum_encoding_str } }
846   \bool_new:N \l__zhnum_active_char_bool
847   \bool_set_true:N \l__zhnum_active_char_bool
848 \fi:

```

encoding 宏包设置选项。

```

style
null
reset
849 \keys_define:nn { zhnum / options }
850 {
851   encoding .choices:nn =
852   { UTF8 , GBK , Big5 }
853   {
854     \str_set:Nx \l__zhnum_encoding_str
855     { \str_fold_case:V \l_keys_choice_tl }
856     \zhnum_load_cfg:o { \l__zhnum_encoding_str }
857   } ,
858   encoding .default:n = { GBK } ,
859   encoding / Bg5 .meta:n = { encoding = Big5 } ,
860   encoding / unknown .code:n =
861   { \msg_error:nnn { zhnumber } { encoding-invalid } {#1} } ,
862   style .multichoice: ,
863   style / Normal .code:n =

```

```

864     {
865         \bool_set_false:N \l__zhnum_ancient_bool
866         \bool_set_true:N \l__zhnum_normal_bool
867     } ,
868     style / Financial .code:n =
869     {
870         \bool_set_false:N \l__zhnum_ancient_bool
871         \bool_set_false:N \l__zhnum_normal_bool
872     } ,
873     style / Ancient .code:n =
874     {
875         \bool_set_true:N \l__zhnum_ancient_bool
876         \bool_set_true:N \l__zhnum_normal_bool
877     } ,
878     style / Simplified .code:n = { \bool_set_true:N \l__zhnum_simp_bool } ,
879     style / Traditional .code:n = { \bool_set_false:N \l__zhnum_simp_bool } ,
880     style .default:n = { Normal , Simplified } ,
881     null .bool_set:N = \l__zhnum_null_bool ,
882     time .choice: ,
883     time / Chinese .code:n = { \bool_set_true:N \l__zhnum_time_bool } ,
884     time / Arabic .code:n = { \bool_set_false:N \l__zhnum_time_bool } ,
885     time .default:n = { Arabic } ,
886     reset .code:n = { \zhnum_reset_config: } ,
887     activechar .bool_set:N = \l__zhnum_active_char_bool ,
888     ganzhi-cyclic .choice: ,
889     ganzhi-cyclic / true .code:n =
890     { \cs_set_eq:NN \zhnum_ganzhi:nnn \zhnum_ganzhi_cyclic:nnn } ,
891     ganzhi-cyclic / false.code:n =
892     { \cs_set_eq:NN \zhnum_ganzhi:nnn \zhnum_ganzhi_normal:nnn } ,
893     ganzhi-cyclic .default:n = { true } ,
894     arabicsep .tl_set:N = \l__zhnum_arabic_sep_tl
895 }
896 \str_new:N \l__zhnum_encoding_str
897 \msg_new:nnnn { zhnumber } { encoding-invalid }
898 { The~encoding~`#1'~is~invalid. }
899 { Available~encodings~are~`UTF8',~`GBK'~and~`Big5'. }

```

`\zhnumsetup` 在文档中设置 `zhnumber` 的接口。

```

900 \NewDocumentCommand \zhnumsetup { +m }
901 {
902     \keys_set:nn { zhnum / options } {#1}
903     \tex_ignorespaces:D
904 }

```

初始化设置和执行宏包选项。

```

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

```

如果没有选定编码,则根据引擎自动设置编码。

```

907 \str_if_empty:NT \l__zhnum_encoding_str
908 {
909     \zhnum_if_unicode_engine:TF
910     { \keys_set:nn { zhnum / options } { encoding = UTF8 } }
911     { \keys_set:nn { zhnum / options } { encoding = GBK } }
912 }
913 </package>

```

第 4 节 中文数字配置文件

```

914 <*config>
915 <!*big5>
916 \zhnum_set_digits_map:nnn { minus } { simp } { 负 }
917 \zhnum_set_digits_map:nnn { minus } { trad } { 負 }
918 </!big5>

```

```

919 <*big5>
920 \zhnum_set_digits_map:nn { minus } { 負 }
921 </big5>
922 \zhnum_set_digits_map:nn { 0 } { 零 }
923 <!*big5>
924 \zhnum_set_digits_map:nn { null } { 〇 }
925 </!big5>
926 <*big5>
927 \zhnum_set_digits_map:nn { null } { 〇 }
928 </big5>
929 \zhnum_set_digits_map:nn { 1 } { 一 }
930 \zhnum_set_digits_map:nn { 2 } { 二 }
931 \zhnum_set_digits_map:nn { 3 } { 三 }
932 \zhnum_set_digits_map:nn { 4 } { 四 }
933 \zhnum_set_digits_map:nn { 5 } { 五 }
934 \zhnum_set_digits_map:nn { 6 } { 六 }
935 \zhnum_set_digits_map:nn { 7 } { 七 }
936 \zhnum_set_digits_map:nn { 8 } { 八 }
937 \zhnum_set_digits_map:nn { 9 } { 九 }
938 \zhnum_set_digits_map:nn { 10 } { 十 }
939 \zhnum_set_digits_map:nn { 100 } { 百 }
940 \zhnum_set_digits_map:nn { 1000 } { 千 }
941 \zhnum_set_digits_map:nn { 20 } { 廿 }
942 \zhnum_set_digits_map:nn { 30 } { 卅 }
943 \zhnum_set_digits_map:nn { 40 } { 卌 }
944 \zhnum_set_digits_map:nn { 200 } { 兩 }
945 <!*big5>
946 \zhnum_set_digits_map:nnn { dot } { simp } { 点 }
947 \zhnum_set_digits_map:nnn { dot } { trad } { 點 }
948 </big5>
949 <*big5>
950 \zhnum_set_digits_map:nn { dot } { 點 }
951 </big5>
952 \zhnum_set_digits_map:nn { and } { 又 }
953 \zhnum_set_digits_map:nn { parts } { 分之 }
954 <!*big5>
955 \zhnum_set_digits_map:nnn { s1 } { simp } { 万 }
956 \zhnum_set_digits_map:nnn { s1 } { trad } { 萬 }
957 \zhnum_set_digits_map:nnn { s2 } { simp } { 亿 }
958 \zhnum_set_digits_map:nnn { s2 } { trad } { 億 }
959 </!big5>
960 <*big5>
961 \zhnum_set_digits_map:nn { s1 } { 萬 }
962 \zhnum_set_digits_map:nn { s2 } { 億 }
963 </big5>
964 \zhnum_set_digits_map:nn { s3 } { 兆 }
965 \zhnum_set_digits_map:nn { s4 } { 京 }
966 \zhnum_set_digits_map:nn { s5 } { 垓 }
967 \zhnum_set_digits_map:nn { s6 } { 秭 }
968 \zhnum_set_digits_map:nn { s7 } { 穰 }
969 <!*big5>
970 \zhnum_set_digits_map:nnn { s8 } { simp } { 沟 }
971 \zhnum_set_digits_map:nnn { s8 } { trad } { 溝 }
972 \zhnum_set_digits_map:nnn { s9 } { simp } { 涧 }
973 \zhnum_set_digits_map:nnn { s9 } { trad } { 澗 }
974 </!big5>
975 <*big5>
976 \zhnum_set_digits_map:nn { s8 } { 溝 }
977 \zhnum_set_digits_map:nn { s9 } { 澗 }
978 </big5>
979 \zhnum_set_digits_map:nn { s10 } { 正 }
980 <!*big5>
981 \zhnum_set_digits_map:nnn { s11 } { simp } { 载 }
982 \zhnum_set_digits_map:nnn { s11 } { trad } { 載 }
983 </!big5>
984 <*big5>
985 \zhnum_set_digits_map:nn { s11 } { 載 }

```

```

986 </big5>
987 \zhnum_set_digits_map:nn { year } { 年 }
988 \zhnum_set_digits_map:nn { month } { 月 }
989 \zhnum_set_digits_map:nn { day } { 日 }
990 <!*big5>
991 \zhnum_set_digits_map:nnn { hour } { simp } { 时 }
992 \zhnum_set_digits_map:nnn { hour } { trad } { 時 }
993 </!big5>
994 <*big5>
995 \zhnum_set_digits_map:nn { hour } { 時 }
996 </big5>
997 \zhnum_set_digits_map:nn { minute } { 分 }
998 \zhnum_set_digits_map:nn { weekday } { 星期 }
999 \zhnum_set_financial_map:nn { null } { 零 }
1000 \zhnum_set_financial_map:nn { 0 } { 零 }
1001 \zhnum_set_financial_map:nn { 1 } { 壹 }
1002 \zhnum_set_financial_map:nn { 2 } { 貳 }
1003 <!*big5>
1004 \zhnum_set_financial_map:nnn { 3 } { simp } { 叁 }
1005 \zhnum_set_financial_map:nnn { 3 } { trad } { 參 }
1006 </!big5>
1007 <*big5>
1008 \zhnum_set_financial_map:nn { 3 } { 參 }
1009 </big5>
1010 \zhnum_set_financial_map:nn { 4 } { 肆 }
1011 \zhnum_set_financial_map:nn { 5 } { 伍 }
1012 <!*big5>
1013 \zhnum_set_financial_map:nnn { 6 } { simp } { 陆 }
1014 \zhnum_set_financial_map:nnn { 6 } { trad } { 陸 }
1015 </!big5>
1016 <*big5>
1017 \zhnum_set_financial_map:nn { 6 } { 陸 }
1018 </big5>
1019 \zhnum_set_financial_map:nn { 7 } { 柒 }
1020 \zhnum_set_financial_map:nn { 8 } { 捌 }
1021 \zhnum_set_financial_map:nn { 9 } { 玖 }
1022 \zhnum_set_financial_map:nn { 10 } { 拾 }
1023 \zhnum_set_financial_map:nn { 100 } { 佰 }
1024 \zhnum_set_financial_map:nn { 1000 } { 仟 }
1025 \zhnum_set_tiangang_map:nn { 1 } { 甲 }
1026 \zhnum_set_tiangang_map:nn { 2 } { 乙 }
1027 \zhnum_set_tiangang_map:nn { 3 } { 丙 }
1028 \zhnum_set_tiangang_map:nn { 4 } { 丁 }
1029 \zhnum_set_tiangang_map:nn { 5 } { 戊 }
1030 \zhnum_set_tiangang_map:nn { 6 } { 己 }
1031 \zhnum_set_tiangang_map:nn { 7 } { 庚 }
1032 \zhnum_set_tiangang_map:nn { 8 } { 辛 }
1033 \zhnum_set_tiangang_map:nn { 9 } { 壬 }
1034 \zhnum_set_tiangang_map:nn { 10 } { 癸 }
1035 \zhnum_set_dizhi_map:nn { 1 } { 子 }
1036 \zhnum_set_dizhi_map:nn { 2 } { 丑 }
1037 \zhnum_set_dizhi_map:nn { 3 } { 寅 }
1038 \zhnum_set_dizhi_map:nn { 4 } { 卯 }
1039 \zhnum_set_dizhi_map:nn { 5 } { 辰 }
1040 \zhnum_set_dizhi_map:nn { 6 } { 巳 }
1041 \zhnum_set_dizhi_map:nn { 7 } { 午 }
1042 \zhnum_set_dizhi_map:nn { 8 } { 未 }
1043 \zhnum_set_dizhi_map:nn { 9 } { 申 }
1044 \zhnum_set_dizhi_map:nn { 10 } { 酉 }
1045 \zhnum_set_dizhi_map:nn { 11 } { 戌 }
1046 \zhnum_set_dizhi_map:nn { 12 } { 亥 }
1047 </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 718, 738

B

bingint internal commands:
 _bingint_read_do:nn 7
bool commands:
 _bool_if:NTF 214, 227, 359, 445, 627, 647, 664, 678, 816, 841
 _bool_lazy_all:nTF 241
 _bool_lazy_and:nnTF 232, 249
 _bool_lazy_any:nTF 792
 _bool_new:N 785, 846
 _bool_set_false:N 759, 865, 870, 871, 879, 884
 _bool_set_true:N 630, 847, 866, 875, 876, 878, 883
 _c_false_bool 219, 220, 803
 _c_true_bool 204, 219, 220, 799
\BooleanFalse 297
\BooleanTrue 295

C

char commands:
 _char_set_catcode_active:n 831
clist commands:
 _clist_map_function:nN 471
 _clist_map_inline:nn 572, 580, 585, 699
cs commands:
 _cs:w 337
 _cs_end: 345
 _cs_generate_variant:Nn
 .. 35, 111, 188, 207, 223, 298, 371, 463, 510, 512, 537, 747
 _cs_if_exist_use:N 487
 _cs_if_exist_use:NTF 458, 493
 _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, 273, 299, 307, 319, 330, 335, 353, 363, 365,
 372, 374, 386, 394, 400, 405, 418, 430, 434, 449, 454,
 456, 461, 484, 489, 500, 513, 515, 517, 519, 521, 703, 705
 _cs_new_eq:NN 466,
 511, 716, 717, 722, 723, 726, 799, 800, 803, 804, 807, 813
 _cs_new_nopar:Npn . 112, 279, 294, 296, 355, 357, 437, 443
 _cs_new_protected:Npn
 475, 599, 601, 606, 608, 613, 615,
 633, 639, 658, 701, 740, 748, 755, 769, 776, 781, 808, 835
 _cs_new_protected_nopar:Npn
 621, 653, 674, 682, 815, 817, 844

\cs_new_protected_nopar:Npx 727
\cs_set:Npn 473
\cs_set_eq:NN 724, 739, 890, 892

D

\DeclareExpandableDocumentCommand . 13, 74, 259, 280, 347

E

else commands:
 _else: 127, 139, 170, 312, 341, 814
encoding 1, 849
exp commands:
 _exp:w 118, 126, 129, 138, 150, 304, 311, 328
 _exp_after:wN 115, 117, 125, 126, 128,
 130, 131, 137, 138, 140, 141, 149, 169, 171, 191, 194,
 195, 196, 301, 303, 310, 311, 313, 314, 315, 316, 325, 327
 _exp_args:Nf 175, 177, 704
 _exp_args:No 637
 _exp_end_continue_f:w
 118, 126, 129, 138, 150, 304, 311, 328
 _exp_not:N 124, 136,
 168, 309, 325, 434, 546, 547, 548, 549, 550, 551, 552,
 558, 559, 560, 561, 562, 564, 570, 576, 577, 583, 592,
 665, 666, 671, 685, 687, 689, 691, 693, 695, 697, 700,
 709, 712, 713, 721, 729, 730, 731, 732, 733, 734, 735, 736
 _exp_not:n 597,
 647, 648, 664, 678, 679, 685, 687, 689, 691, 693, 695, 697
 _exp_stop_f: 119, 168, 193, 198, 305, 376

F

fi commands:
 _fi: ... 124, 132, 142, 172, 197, 309, 317, 325, 343, 384, 848
file commands:
 _file_if_exist:nTF 757
 _file_input:n 763

G

ganzhi-cyclic 4
group commands:
 _group_begin: 22, 83, 268, 289, 538, 761
 _group_end: 25, 86, 271, 292, 596, 765

H

hundred commands:
 _c_one_hundred 413

I

if commands:
 _if:w 124, 136, 309, 325
 _if_case:w 193, 376
 _if_int_compare:w 168, 339
 _if_predicate:w 806

`\IfBooleanTF` 340, 350
`\IfNoValueTF` 15, 76, 261, 282, 472
int commands:
`\c_eleven` 465, 479
`\c_five` 251, 426
`\c_four` 177, 184, 192, 412, 425
`\int_compare:nNnTF` 100, 103,
158, 161, 180, 183, 211, 217, 226, 229, 230, 238, 239,
257, 332, 388, 396, 402, 486, 491, 502, 505, 523, 526, 706
`\int_compare_p:n` 251
`\int_compare_p:nNn` 234, 243, 244
`\int_div_truncate:nn` 192, 410, 412, 413, 414, 423, 425, 440
`\int_eval:n` 101, 106, 155,
184, 221, 279, 403, 479, 507, 514, 516, 518, 520, 531, 563
`\int_if_exist:NTF` 90, 275
`\int_incr:N` 477, 481
`\int_mod:nn` ... 177, 407, 420, 441, 462, 496, 524, 531, 704
`\int_new:N` 464, 833, 834
`\int_set:Nn` 821, 822, 824, 826, 827
`\int_set_eq:NN` 465
`\int_step_function:nnnN` 829
`\int_step_inline:nnnn` 554, 567, 707
`\int_to_arabic:n` 361, 447
`\int_zero:N` 470
`\c_nine` 168
`\c_one`
. 147, 161, 184, 221, 243, 251, 332, 397, 410, 423, 486, 830
`\c_seven` 416, 428
`\c_six` 413
`\c_ten` 402, 410, 423
`\c_three` 396
`\l_tmpa_int` 470, 477, 479
`\c_twelve` 397
`\c_two` 180, 234
`\c_zero` ... 100, 103, 120, 158, 183, 211, 217, 226, 229,
230, 238, 239, 244, 257, 339, 491, 502, 505, 523, 526, 706

K

keys commands:
`\l_keys_choice_tl` 855
`\keys_define:nn` 597, 849
`\keys_set:nn` 23, 84, 269, 290, 902, 905, 910, 911

M

msg commands:
`\msg_error:nn` 11
`\msg_error:nnn` 767, 861
`\msg_expandable_error:nnn` 95
`\msg_new:nnn` 3, 96
`\msg_new:nnnn` 786, 897

N

`\NewDocumentCommand` 20, 81, 266, 287, 468, 900
`null` 3, 849

O

or commands:
`\or:` 194, 195, 196, 378, 379, 380, 381, 382, 383

P

prg commands:
`\prg_do_nothing:` 807
`\ProcessKeysOptions` 906
prop commands:
`\prop_clear:N` 778
`\prop_get:NnNTF` 636, 641, 643, 660
`\prop_gset_eq:NN` 782
`\prop_if_exist:NTF` 750
`\prop_map_function:NN` 623, 624, 655
`\prop_map_inline:Nn` 720
`\prop_new:N` 617, 618, 619, 620, 779
`\prop_put:Nnn` 600, 604, 607, 611, 614, 616
`\prop_put_if_new:Nnn` 603, 610
`\prop_set_eq:NN` 753

Q

quark commands:
`\q_mark` 40, 42
`\q_nil` 8, 28, 32, 40, 205
`\quark_if_nil:nTF` 31, 38, 44
`\quark_if_recursion_tail_stop:N` 210, 324
`\quark_if_recursion_tail_stop_do:Nn` 148
`\q_recursion_stop` 119, 144, 150, 205, 305
`\q_recursion_tail` 8, 119, 205, 305
`\q_stop` 28, 29, 32, 36, 40, 42, 349, 351, 353, 373, 374, 431, 434

R

`\RequirePackage` 12
`reset` 4, 849

S

str commands:
`\c_colon_str` 434
`\str_case:nnTF` 819
`\str_fold_case:n` 855
`\str_if_empty:NTF` 907
`\str_if_eq:NNTF` 743
`\str_new:N` 783, 784, 896
`\str_set:Nn` 810, 837, 838, 854
`\str_set_eq:NN` 751, 811
`style` 3, 849
sys commands:
`\sys_if_engine luatex_p:` 795
`\sys_if_engine uptex_p:` 796
`\sys_if_engine xetex_p:` 794

T

T_EX and L^AT_EX₂_ε commands:
`\ifpackagelater` 10
`\@zhdig` 279
`\@zhnum` 112
`\pagenumbering` 2, 2
`\tiangan` 4
`\zhdate` 2, 2
`\zhdig` 1, 2, 4
`\zhdigits` 1, 1, 1, 3, 3, 4
`\zh dizhi` 2

\zhganzhi	2	\zhnum_assgin_ganzhi:nn	624, 701
\zhganzhinian	3	\zhnum_blank_to_zero:n	46, 48, 56, 58, 63, 68
\zhnum	1, 2, 4	\zhnum_check_financial:nn	621
\zhnumber	1, 1, 3, 3, 4	\zhnum_check_simp:nn	621
\zhnumExtendScaleMap	3, 3	\zhnum_counter:n	77, 85, 88
\zhnumsetup	1, 3, 4	\zhnum_decimal:nn	33, 61
\zhtiangan	2	\zhnum_digit_map:n	181, 228, 235, 236, 252, 253, 257, 454
\zhtime	2	\zhnum_digits:Nn	283, 291, 295, 297, 299
\zhweekday	2	\zhnum_digits_counter:n	262, 270, 273
tex commands:		\zhnum_digits_null:n	276, 279, 294, 360
\tex_day:D	356	\zhnum_digits_zero:n	66, 294
\tex_ignorespaces:D	903	\zhnum_ganzhi:nnn	511, 514, 516, 518, 890, 892
\tex_month:D	356	\zhnum_ganzhi_cyclic:nnn	489, 890
\tex_number:D	116, 192, 302	\zhnum_ganzhi_nian:n	520, 521
\tex_time:D	440, 441	\zhnum_ganzhi_normal:nnn	484, 511, 892
\tex_year:D	356	\zhnum_if_digit:NTF	146, 166, 321
time	3	\zhnum_if_unicode_engine:TF	792, 909
tl commands:		\zhnum_if_unicode_engine_p:	792, 806
\tl_const:Nn	467, 700, 709, 717, 721	\zhnum_input_cfg:n	752, 755
\tl_count:n	175	\zhnum_int:N	91
\tl_if_blank:NTF	51, 64, 70	\zhnum_int:n	88, 112, 360, 446
\tl_if_exist:NTF	480	\zhnum_integer:n	10, 39, 113
\tl_put_right:Nn	544, 556, 569, 574, 582, 591	\zhnum_load_cfg:n	740, 845, 856
\tl_set:Nn	478, 482, 539, 644, 645,	\zhnum_number:n	16, 24, 27, 53, 72
	651, 676, 684, 686, 688, 690, 692, 694, 696, 702, 710, 724	\zhnum_parse_config:	621, 745, 813
\tl_set_eq:NN	739	\zhnum_parse_number:n	101, 106, 174
\l_tmpa_tl	478, 480, 482, 539,	\zhnum_parse_number:nn	163, 174
	544, 556, 569, 574, 582, 591, 597, 636, 637, 641, 648, 660	\zhnum_process_number:NNNNNN	215, 224
\l_tmpb_tl	643, 644, 648	\zhnum_reset_config:	806, 886
tl internal commands:		\zhnum_scale_map:n	216, 456
_tl_act:NNNnn	7	\zhnum_scale_map_hook:n	459, 466, 473
\TrimSpaces	468	\zhnum_scale_map_loop:n	456
U		\zhnum_set_active:	816, 817
use commands:		\zhnum_set_alias:	656, 726
\use:N	455, 503, 506, 507, 524, 528	\zhnum_set_alias:NN	726, 729, 731, 733, 735, 739
\use:n	118, 126, 138, 150, 194, 304, 311, 328, 432, 594	\zhnum_set_catcode:	762, 806
\use_i:nn	169, 212, 800	\zhnum_set_cfg_name:Nn	742, 806
\use_i:nnn	196	\zhnum_set_digits_map:nn	
\use_i_ii:nnn	195		599, 920, 922, 924, 927, 929, 930,
\use_ii:nn	171, 804		931, 932, 933, 934, 935, 936, 937, 938, 939, 940, 941,
\use_none:n	252		942, 943, 944, 950, 952, 953, 961, 962, 964, 965, 966,
			967, 968, 976, 977, 979, 985, 987, 988, 989, 995, 997, 998
Z		\zhnum_set_digits_map:nnn	599, 916, 917, 946, 947,
\zhcurrtime	2, 437		955, 956, 957, 958, 970, 971, 972, 973, 981, 982, 991, 992
\zhdate	2, 347	\zhnum_set_dizhi_map:nn	599, 1035, 1036,
\zhdig	2, 4, 259		1037, 1038, 1039, 1040, 1041, 1042, 1043, 1044, 1045, 1046
\zhdigits	1, 4, 280	\zhnum_set_financial_map:nn	
\zhdigitsoptions	280		599, 999, 1000, 1001, 1002,
\zhdigwithoptions	263, 266		1008, 1010, 1011, 1017, 1019, 1020, 1021, 1022, 1023, 1024
\zhdizhi	2, 515	\zhnum_set_financial_map:nnn	
\zhganzhi	2, 517		599, 1004, 1005, 1013, 1014
\zhganzhinian	3, 519	\zhnum_set_scale:n	471, 475
\zhnum	2, 4, 74	\zhnum_set_tiangang_map:nn	599,
zhnum commands:			1025, 1026, 1027, 1028, 1029, 1030, 1031, 1032, 1033, 1034
\zhnum_assgin_const:	629, 653	\zhnum_set_week_day:	621
\zhnum_assgin_const_tl:Nn	662, 670, 717, 724	\zhnum_set_zero:	621

\zhnum_split_number:nn	184, 201, 202	_zhnum_output:nnwnn	147, 152
\zhnum_split_number:NNnNNNNw	204, 208, 219, 220	_zhnum_output_digits:NN	322, 335
\zhnum_two_digits:n	386	_zhnum_parse_number:nnn	177, 178
\zhnum_update_cfg:n	744, 748	\c_zhnum_parts_tl	47, 57
\zhnum_Zeller:nnn	376, 386	_zhnum_prop_gset_eq:Nn	764, 781
\zhnum_Zeller_aux:Nnnn	386	_zhnum_prop_initial:Nn	760, 776
\zhnum_Zeller_Gregorian:nnn	390, 405	_zhnum_read_abs_loop:Nw	140, 144
\zhnum_Zeller_Julian:nnn	391, 418	_zhnum_read_digits:w	301, 330
\zhnum_zero_mod:nn	703, 712, 713	_zhnum_read_digits_loop:NN	314, 319, 327
zhnum internal commands:			
\l_zhnum_active_char_bool	816, 841, 846, 847, 887	_zhnum_read_integer:www	115, 156
\l_zhnum_ancient_bool	233, 250, 865, 870, 875	_zhnum_read_sign_loop:N	117, 122, 125
\c_zhnum_and_tl	54	_zhnum_read_sign_loop:NN	303, 307, 310
\l_zhnum_arabic_sep_tl	361, 447, 894	_zhnum_read_zeros_loop:N	130, 134, 137
\l_zhnum_byte_max_int	824, 827, 831, 834	\l_zhnum_reset_bool	627, 630, 759, 785
\l_zhnum_byte_min_int	821, 822, 826, 830, 833	_zhnum_result:nn	120, 152, 153, 154
\l_zhnum_cfg_map_finan_prop	599, 636, 660, 773	\c_zhnum_sat_tl	377
\l_zhnum_cfg_map_ganzhi_prop	599, 624, 720, 774	\l_zhnum_sat_tl	694
\l_zhnum_cfg_map_prop	599, 623, 655, 771	\l_zhnum_scale_int	462, 464, 465, 481
\l_zhnum_cfg_map_var_prop	599, 641, 643, 772	\l_zhnum_simp_bool	647, 878, 879
\l_zhnum_cfg_str	742, 743, 750, 751, 771, 772, 773, 774, 783	_zhnum_split_number_aux:nnn	185, 189
_zhnum_check_simp_aux:nn	635, 637, 639	_zhnum_split_number_aux:wnn	191, 200
_zhnum_counter_error:n	92, 94, 277	\c_zhnum_sun_tl	378
_zhnum_date:www	349, 353	\l_zhnum_sun_tl	696
_zhnum_date_aux:nnn	354, 356, 357	\c_zhnum_ten_tl	255, 731
_zhnum_date_aux:Nnnnn	361, 363	\c_zhnum_thousand_tl	228, 735
_zhnum_date_aux:NNnnnn	360, 364, 365	\c_zhnum_thu_tl	382
\c_zhnum_day_tl	369	\l_zhnum_thu_tl	690
\l_zhnum_day_tl	697	_zhnum_time:ww	431, 434
\c_zhnum_dot_tl	63, 325	_zhnum_time_aux:nn	436, 439, 443
\l_zhnum_encoding_str	810, 811, 819, 837, 840, 845, 854, 856, 896, 907	_zhnum_time_aux:Nnnn	443
_zhnum_fraction:www	40, 42	\l_zhnum_time_bool	359, 445, 883, 884
\c_zhnum_fri_tl	383	\c_zhnum_tue_tl	380
\l_zhnum_fri_tl	692	\l_zhnum_tue_tl	686
_zhnum_ganzhi_cyclic_mod:nnnn	489, 495	_zhnum_update_cfg_prop:N	753, 760, 764, 769
\c_zhnum_hour_tl	451	\c_zhnum_wed_tl	381
\c_zhnum_hundred_tl	236, 733	\l_zhnum_wed_tl	688
_zhnum_integer_or_fraction:www	32, 36	_zhnum_week_day:www	351, 373, 374
\l_zhnum_last_cfg_str	743, 751, 784	\c_zhnum_weekday_tl	685, 687, 689, 691, 693, 695, 697
_zhnum_loop_end:wnn	148, 154	\c_zhnum_year_tl	367
\c_zhnum_minus_tl	105, 162, 333	_zhnum_Zeller_aux:Nnnn	390, 391, 394
\l_zhnum_minus_tl	541	_zhnum_zero_mod_aux:nn	704, 705
\c_zhnum_minute_tl	452	\c_zhnum_zero_tl	65, 71, 108, 159, 214, 227, 230, 239, 729
\c_zhnum_mon_tl	379	zhnumber	1, 4, 13
\l_zhnum_mon_tl	684	zhnumberwithoptions	17, 20, 74
\c_zhnum_month_tl	368	zhnumExtendScaleMap	3, 468
\l_zhnum_normal_bool	664, 866, 871, 876	zhnumsetup	3, 900
\l_zhnum_null_bool	678, 881	zhnumwithoptions	78, 81
\l_zhnum_null_tl	542, 679	zhtiangang	2, 513
_zhnum_number:www	27	zhtime	2, 430
		zhtoday	2, 355
		zhweekday	2, 372