

# zhnumber 宏包

李清

sobenlee@gmail.com

2015/05/21 v2.2\*

## 1 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, 它提供的三个格式转换命令 `\zhnumber`, `\zhdigits` 和 `\zhnum` 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<sup>A</sup>T<sub>E</sub>X 3 项目的 `expl3`, `xparse` 和 `l3keys2e` 宏包。

## 2 使用方法

---

`encoding`

---

Updated: 2014-09-09

---

`encoding = <GBK|Big5|UTF8>`

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 `upLATEX`, `XYLATEX` 和 `LuaLATEX`, 不用指定编码, 宏包将自动使用 UTF8 编码。只有 `LATEX` 和 `pdfLATEX` 需要指定编码, 如果没有指定, 默认将使用 GBK。

---

`\zhnumber`

☆

`\zhnumber <{number}>`

---

Updated: 2014-09-12

---

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`

☆

`\zhdigits <{number}>`

`\zhdigits * <{number}>`

---

Updated: 2014-09-09

---

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  
二零一二二零二零一二零

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

---

`\zhnum`

☆

`\zhnum <{counter}>`

---

Updated: 2014-09-09

---

与 `\roman` 等类似, 用于将 L<sup>A</sup>T<sub>E</sub>X 计数器的值转换为中文数字。例如

二

```
1 \zhnum{section}
```

`\zhweekday` ☆ `\zhweekday {⟨yyyy/mm/dd⟩}`

New: 2012-05-25

输出日期当天的星期。例如  
星期日

```
1 \zhweekday{2012/5/20}
```

`\zhdate` ☆ `\zhdate {⟨yyyy/mm/dd⟩}`  
`\zhdate * {⟨yyyy/mm/dd⟩}`

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  
2012 年 5 月 21 日星期一

```
1 \zhdate{2012/5/21}\  
2 \zhdate*{2012/5/21}
```

`\zhtoday` ☆ 与 `\today` 类似, 以中文输出当天的日期。例如

New: 2012-05-25

2015 年 6 月 19 日

```
1 \zhtoday
```

`\zhtime` ☆ `\zhtime {⟨hh:mm⟩}`

New: 2012-05-25

以中文格式输出时间。例如

23 时 56 分

```
1 \zhtime{23:56}
```

`\zhcurrtime` ☆ 输出当前的时间。例如

New: 2012-05-25

18 时 27 分

```
1 \zhcurrtime
```

`\zhtiangan` ☆ `\zhtiangan {⟨number⟩}`

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 \zhtiangan{1} \zhtiangan{2} \zhtiangan{3}  
2 \zhtiangan{4} \zhtiangan{5} \zhtiangan{10}
```

`\zhdizhi` ☆ `\zhdizhi {⟨number⟩}`

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 \zhdizhi{1} \zhdizhi{2} \zhdizhi{3}  
2 \zhdizhi{4} \zhdizhi{5} \zhdizhi{12}
```

`\zhganzhi` ☆ `\zhganzhi {⟨number⟩}`

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  
丁卯 戊辰 癸亥

```
1 \zhganzhi{1} \zhganzhi{2} \zhganzhi{3} \  
2 \zhganzhi{4} \zhganzhi{5} \zhganzhi{60}
```

`\zhganzhinian` ☆ `\zhganzhinian {⟨year⟩}`

New: 2015-05-20

输出公元纪年 ⟨*year*⟩ 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 乙未

```
1 \zhganzhinian{1898} \zhganzhinian{-246} \  
2 \zhganzhinian{-2697} \zhganzhinian{⟨year⟩}
```

`\zhnumExtendScaleMap` `\zhnumExtendScaleMap [⟨character⟩] {⟨character1⟩, ⟨character2⟩, ..., ⟨charactern⟩}`

New: 2012-05-25

缺省状态下 `\zhnumber` 能正确中文格式化的最大整数是  $10^{48} - 1$ , `\zhdigits` 不受这个大小的限制。可以通过 `\zhnumExtendScaleMap` 来扩展 `\zhnumber`。⟨*character<sub>i</sub>*⟩ 设置  $10^{4(i+1)}$ 。若给出可选项 ⟨*character*⟩, 则当数字大于  $10^{4(n+12)} - 1$  时, 统一用 ⟨*character*⟩ 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 = ⟨<b>Arabic</b>   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

二〇一五年六月十九日十八时二十七分

1 `\zhnumsetup{time=Chinese}`

2 `\zhtoday\zhcurrtime`

<code>style</code>	<code>style = ⟨<b>Simplified</b>   Traditional   Normal   Financial   Ancient⟩</code>
--------------------	---

Updated: 2012-05-25 意义分别为

<code>Simplified</code>	以简体中文输出数字(对 Big5 编码无效);
<code>Traditional</code>	以繁体中文输出数字(对 Big5 编码无效);
<code>Normal</code>	以小写形式输出中文数字;
<code>Financial</code>	以大写形式输出中文数字;
<code>Ancient</code>	以廿输出 20,以卅输出 30,以卌输出 40,以𠫪输出 200。

可以设置 `style` 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

1 `\zhnumsetup{style={Traditional,Financial}}`

2 `\zhnumber{62012.3}\`

3 `\zhnumsetup{style=Ancient}`

4 `\zhnumber{21}`

<code>null</code>	<code>null = ⟨true   false⟩</code>
-------------------	------------------------------------

缺省状态下,除了 `\zhdigits` 外,其它的格式转换命令,将 0 映射成零,如果需要将 0 映射成〇,可以使用这个选项。

<code>ganzhi-cyclic</code>	<code>ganzhi-cyclic = ⟨true   false⟩</code>
----------------------------	---

New: 2015-05-20 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”,  $E_n$  设置  $10^n$ ,  $F_n$  设置数字  $n$  的大写,  $T_n$  设置数字  $n$  的天干,  $D_n$  设置数字  $n$  的地支,而  $GZ_n$  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。

<code>reset</code>	<code>reset</code>
Updated: 2014-09-12	用于恢复 <code>zhnumber</code> 对阿拉伯数字的初始化映射。 <code>zhnumber</code> 的中文数字初始化设置见源代码(第 4 节)。
<code>activechar</code>	<code>activechar = &lt;true false&gt;</code>
New: 2014-09-09	在 <code>L<sup>A</sup>T<sub>E</sub>X</code> 或者 <code>pdfL<sup>A</sup>T<sub>E</sub>X</code> 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, <code>zhnumber</code> 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 <code>encoding</code> 或者 <code>reset</code> 选项才会有效果。
<code>\zhnumber</code>	<code>\zhnumber    [&lt;options&gt;] {&lt;number&gt;}</code>
<code>\zhdigits</code>	<code>\zhdigits * [&lt;options&gt;] {&lt;number&gt;}</code>
<code>\zhnum</code>	<code>\zhnum       [&lt;options&gt;] {&lt;counter&gt;}</code>
Updated: 2014-09-09	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 <code>&lt;options&gt;</code> 与 <code>\zhnumsetup</code> 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。

### 3 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 \@ifpackagelater { expl3 } { 2014/08/25 } { }
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<sup>A</sup>T<sub>E</sub>X 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<sup>A</sup>T<sub>E</sub>X 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:n { '#1' is not a LATEX counter. } }
96 \cs_new:Npn \zhnum_int:n #1
97 {
98   \int_compare:nNnTF {#1} > \c_zero
99   { \zhnum_parse_number:f { \int_eval:n {#1} } }
100   {
101     \int_compare:nNnTF {#1} < \c_zero
102     {
103       \c__zhnum_minus_tl
104       \zhnum_parse_number:f { \int_eval:n { - #1 } }
105     }
106     { \c__zhnum_zero_tl }
107   }
108 }
109 \cs_generate_variant:Nn \zhnum_int:n { c }

```

(End definition for `\zhnum_counter:n` and `\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。

```

110 \cs_new:Npn \zhnum_integer:n #1
111 {
112   \exp_after:wN \__zhnum_read_integer:www
113   \tex_number:D
114   \exp_after:wN \__zhnum_read_sign_loop:N
115   \tex_romannumeral:D -`0 \use:n
116   #1 \exp_stop_f: \q_recursion_tail \q_recursion_stop
117   \__zhnum_result:nn { \c_zero } { } ;
118 }
119 \cs_new:Npn \__zhnum_read_sign_loop:N #1
120 {
121   \if:w + \if:w - \exp_not:N #1 + \fi: \exp_not:N #1
122   \exp_after:wN \__zhnum_read_sign_loop:N
123   \tex_romannumeral:D -`0 \exp_after:wN \use:n
124   \else:
125     1 \exp_after:wN ;
126     \tex_romannumeral:D -`0
127     \exp_after:wN \__zhnum_read_zeros_loop:N
128     \exp_after:wN #1
129   \fi:
130 }
131 \cs_new:Npn \__zhnum_read_zeros_loop:N #1
132 {
133   \if:w 0 \exp_not:N #1
134   \exp_after:wN \__zhnum_read_zeros_loop:N
135   \tex_romannumeral:D -`0 \exp_after:wN \use:n
136   \else:
137     \exp_after:wN \__zhnum_read_abs_loop:Nw
138     \exp_after:wN #1

```

```

139   \fi:
140 }

```

(End definition for \zhnum\_integer:n.)

\\_zhnum\_read\_abs\_loop:Nw 当数字很大时,l3bigit 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。

```

141 \cs_new:Npn \_zhnum_read_abs_loop:Nw #1#2 \q_recursion_stop
142 {
143   \zhnum_if_digit:NTF #1
144     { \_zhnum_output:nnwnn { + \c_one } #1 }
145     { \quark_if_recursion_tail_stop_do:Nn #1 { \_zhnum_loop_end:wnn } }
146   \exp_after:wN \_zhnum_read_abs_loop:Nw
147   \tex_romannumeral:D -`0 \use:n #2 \q_recursion_stop
148 }
149 \cs_new:Npn \_zhnum_output:nnwnn #1#2#3 \_zhnum_result:nn #4#5
150 { #3 \_zhnum_result:nn { #4#1 } { #5#2 } }
151 \cs_new:Npn \_zhnum_loop_end:wnn #1 \_zhnum_result:nn #2#3
152 { \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。

```

153 \cs_new:Npn \_zhnum_read_integer:www #1 ; #2 ; #3 ;
154 {
155   \int_compare:nNnTF {#2} = \c_zero
156     { \c_zhnum_zero_tl }
157     {
158       \int_compare:nNnF {#1} = \c_one
159       { \c_zhnum_minus_tl }
160       \zhnum_parse_number:nn {#2} {#3}
161     }
162 }

```

(End definition for \\_zhnum\_read\_integer:www.)

\zhnum\_if\_digit:NTF 判断 #1 是否为数字位。

```

163 \cs_new:Npn \zhnum_if_digit:NTF #1
164 {
165   \if_int_compare:w \c_nine < 1 \exp_not:N #1 \exp_stop_f:
166   \exp_after:wN \use_i:nn
167   \else:
168   \exp_after:wN \use_ii:nn
169   \fi:
170 }

```

(End definition for \zhnum\_if\_digit:NTF.)

\zhnum\_parse\_number:n  
\zhnum\_parse\_number:nn

```

171 \cs_new:Npn \zhnum_parse_number:n #1
172 { \exp_args:Nf \zhnum_parse_number:nn { \tl_count:n {#1} } {#1} }
173 \cs_new:Npn \zhnum_parse_number:nn #1
174 { \exp_args:Nf \_zhnum_parse_number:nnn { \int_mod:nn {#1} \c_four } {#1} }
175 \cs_new:Npn \_zhnum_parse_number:nnn #1#2
176 {
177   \int_compare:nNnTF {#2} < \c_two
178     { \zhnum_digit_map:n }
179     {
180       \int_compare:nNnTF {#1} = \c_zero
181       { \zhnum_split_number:fn { \int_eval:n { #2 / \c_four - \c_one } } }
182       { \_zhnum_split_number_aux:nnn {#1} {#2} }
183     }
184 }
185 \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。
186 \cs_new:Npn \__zhnum_split_number_aux:nnn #1#2
187 {
188   \exp_after:wN \__zhnum_split_number_aux:wwn
189   \tex_number:D \int_div_truncate:nn {#2} \c_four
190   \if_case:w #1 \exp_stop_f:
191     \or: \exp_after:wN \use:n
192     \or: \exp_after:wN \use_i_ii:nnn
193     \or: \exp_after:wN \use_i:nnn
194   \fi:
195   { \exp_stop_f: ; 0 } 0 0 ;
196 }
197 \cs_new:Npn \__zhnum_split_number_aux:wwn #1 ; #2 ; #3
198 { \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。
199 \cs_new:Npn \zhnum_split_number:nn #1#2
200 {
201   \zhnum_split_number:NNnNNNNw \c_true_bool \c_true_bool {#1}
202   #2 \q_recursion_tail \q_nil \q_nil \q_nil \q_recursion_stop
203 }
204 \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。
205 \cs_new:Npn \zhnum_split_number:NNnNNNNw #1#2#3#4#5#6#7
206 {
207   \quark_if_recursion_tail_stop:N #4
208   \int_compare:nNnTF { #4#5#6#7 } = \c_zero
209     { \use_i:nn }
210     {
211       \bool_if:NF #1 { \c__zhnum_zero_tl }
212       \zhnum_process_number:NNNNNN #4#5#6#7#1#2
213       \zhnum_scale_map:n {#3}
214       \int_compare:nNnTF {#7} = \c_zero
215     }
216     { \zhnum_split_number:NNfNNNNw \c_false_bool \c_true_bool }
217     { \zhnum_split_number:NNfNNNNw \c_true_bool \c_false_bool }
218     { \int_eval:n { #3 - \c_one } }
219   }
220 \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。
221 \cs_new:Npn \zhnum_process_number:NNNNNN #1#2#3#4#5#6
222 {
223   \int_compare:nNnTF {#1} = \c_zero
224     { \bool_if:NF #6 { \c__zhnum_zero_tl } }
225     { \zhnum_digit_map:n {#1} \c__zhnum_thousand_tl }
226   \int_compare:nNnTF {#2} = \c_zero
227     { \int_compare:nNnF { #1 * (#3#4) } = \c_zero { \c__zhnum_zero_tl } }
228     {
229       \bool_if:NTF
230         { \l__zhnum_ancient_bool && \int_compare_p:nNn {#2} = \c_two }
231         { \zhnum_digit_map:n { #2 00 } }
232         { \zhnum_digit_map:n {#2} \c__zhnum_hundred_tl }
233     }
234   \int_compare:nNnTF {#3} = \c_zero
235     { \int_compare:nNnF { #2 * #4 } = \c_zero { \c__zhnum_zero_tl } }
236     {
237       \bool_if:NF
238       {
239         \int_compare_p:nNn {#3} = \c_one &&
240         \int_compare_p:nNn {#1#2} = \c_zero && #6 && #5
241       }

```

```

242     {
243       \bool_if:nTF
244       {
245         \l__zhnum_ancient_bool      &&
246         ( \int_compare_p:nNn {#3} = \c_two ||
247           \int_compare_p:nNn {#3} = \c_three ||
248           \int_compare_p:nNn {#3} = \c_four )
249       }
250       { \zhnum_digit_map:n { #3 0 } \use_none:n }
251       { \zhnum_digit_map:n {#3} }
252     }
253     \c__zhnum_ten_tl
254   }
255   \int_compare:nNnF {#4} = \c_zero { \zhnum_digit_map:n {#4} }
256 }

```

(End definition for \zhnum\_process\_number:NNNNNN.)

**\zhdigits** 将输入的数字输出为中文数字串输出。

```

\zhdigitswithoptions 257 \DeclareExpandableDocumentCommand \zhdigits { +s +o +m }
258 {
259   \IfNoValueTF {#2}
260   { \zhnum_digits:Nn #1 }
261   { \zhdigitswithoptions {#1} {#2} }
262   {#3}
263 }
264 \NewDocumentCommand \zhdigitswithoptions { +m +m +m }
265 {
266   \group_begin:
267   \keys_set:nn { zhnum / options } {#2}
268   \zhnum_digits:Nn #1 {#3}
269   \group_end:
270 }

```

(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 271 \cs_new_nopar:Npn \zhnum_digits_zero:n
272 { \zhnum_digits:Nn \BooleanTrue }
273 \cs_new_nopar:Npn \zhnum_digits_null:n
274 { \zhnum_digits:Nn \BooleanFalse }
275 \cs_generate_variant:Nn \zhnum_digits_null:n { V }

```

(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。

```

276 \cs_new:Npn \zhnum_digits:Nn #1#2
277 {
278   \exp_after:wN \__zhnum_read_digits:w
279   \tex_number:D
280   \exp_after:wN \__zhnum_read_sign_loop:NN \exp_after:wN #1
281   \tex_romannumeral:D -`0 \use:n
282   #2 \exp_stop_f: \q_recursion_tail \q_recursion_stop
283 }
284 \cs_new:Npn \__zhnum_read_sign_loop:NN #1#2
285 {
286   \if:w + \if:w - \exp_not:N #2 + \fi: \exp_not:N #2
287   \exp_after:wN \__zhnum_read_sign_loop:NN \exp_after:wN #1
288   \tex_romannumeral:D -`0 \exp_after:wN \use:n
289   \else:
290     1 \exp_after:wN ;
291     \exp_after:wN \__zhnum_read_digits_loop:NN
292     \exp_after:wN #1
293     \exp_after:wN #2
294   \fi:
295 }
296 \cs_new:Npn \__zhnum_read_digits_loop:NN #1#2
297 {
298   \zhnum_if_digit:NTF #2

```

```

299     { \_zhnum_output_digits:NN #1#2 }
300     {
301       \quark_if_recursion_tail_stop:N #2
302       \if:w .\exp_not:N #2 \exp_after:wN \c_zhnum_dot_tl \fi:
303     }
304     \exp_after:wN \_zhnum_read_digits_loop:NN \exp_after:wN #1
305     \tex_romannumeral:D -`0 \use:n
306   }
307   \cs_new:Npn \_zhnum_read_digits:w #1 ;
308   {
309     \int_compare:nNf {#1} = \c_one
310     { \c_zhnum_minus_tl }
311   }
312   \cs_new:Npn \_zhnum_output_digits:NN #1#2
313   {
314     \cs:w
315     c_zhnum_
316     \if_int_compare:w #2 = \c_zero
317     \IfBooleanTF #1 { zero } { null }
318     \else:
319     #2
320     \fi:
321     _tl
322     \cs_end:
323   }

```

(End definition for \zhnum\_digits:Nn.)

**\zhdate** 输出中文日期。

```

324 \DeclareExpandableDocumentCommand \zhdate { +s +m }
325 {
326   \_zhnum_date:www #2 \q_stop
327   \IfBooleanT #1
328   { \_zhnum_week_day:www #2 \q_stop }
329 }
330 \cs_new:Npn \_zhnum_date:www #1/#2/#3 \q_stop
331 {
332   \zhnum_check_time:Nn \zhnum_digits_null:n {#1} \c_zhnum_year_tl
333   \zhnum_check_time:Nn \zhnum_int:n {#2} \c_zhnum_month_tl
334   \zhnum_check_time:Nn \zhnum_int:n {#3} \c_zhnum_day_tl
335 }

```

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

**\zhtoday** 输出当天日期。

```

336 \cs_new_nopar:Npn \zhtoday
337 {
338   \zhnum_check_time:Nn \zhnum_digits_null:V \tex_year:D \c_zhnum_year_tl
339   \zhnum_check_time:Nn \zhnum_int:n \tex_month:D \c_zhnum_month_tl
340   \zhnum_check_time:Nn \zhnum_int:n \tex_day:D \c_zhnum_day_tl
341 }

```

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

**\zhnum\_check\_time:Nn** 判断是用中文数字还是用阿拉伯数组。

```

342 \cs_new:Npn \zhnum_check_time:Nn #1
343 { \bool_if:NTF \l_zhnum_time_bool {#1} { \int_to_arabic:n } }

```

(End definition for \zhnum\_check\_time:Nn.)

**\zhweekday** 输出星期

```

344 \cs_new:Npn \zhweekday #1
345 { \_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  $d = h + 5 \pmod{7} + 1$ 。

```

346 \cs_new:Npn \_zhnum_week_day:www #1/#2/#3 \q_stop
347 {
348   \if_case:w \zhnum_Zeller:nnn {#1} {#2} {#3} \exp_stop_f:
349     \c_zhnum_sat_tl
350     \or: \c_zhnum_sun_tl
351     \or: \c_zhnum_mon_tl
352     \or: \c_zhnum_tue_tl
353     \or: \c_zhnum_wed_tl
354     \or: \c_zhnum_thu_tl
355     \or: \c_zhnum_fri_tl
356   \fi:
357 }

```

(End definition for \\_zhnum\_week\_day:www.)

\zhnum\_Zeller:nnn 用 Zeller 公式<sup>1</sup> 计算星期几。

```

\zhnum_Zeller_aux:Nnnn
\zhnum_two_digits:n
358 \cs_new:Npn \zhnum_Zeller:nnn #1#2#3
359 {
360   \int_compare:nNnTF
361     { #1 \zhnum_two_digits:n {#2} \zhnum_two_digits:n {#3} } > { 1582 10 04 }
362     { \_zhnum_Zeller_aux:Nnnn \zhnum_Zeller_Gregorian:nnn }
363     { \_zhnum_Zeller_aux:Nnnn \zhnum_Zeller_Julian:nnn }
364     {#1} {#2} {#3}
365   }
366   \cs_new:Npn \_zhnum_Zeller_aux:Nnnn #1#2#3#4
367   {
368     \int_compare:nNnTF {#3} < \c_three
369       { #1 { #2 - \c_one } { #3 + \c_twelve } {#4} }
370       { #1 {#2} {#3} {#4} }
371   }
372   \cs_new:Npn \zhnum_two_digits:n #1
373   {
374     \int_compare:nNnT {#1} < \c_ten { 0 }
375     \int_eval:n {#1}
376   }

```

(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  $Y --$ 。

```

377 \cs_new:Npn \zhnum_Zeller_Gregorian:nnn #1#2#3
378 {
379   \int_mod:nn
380   {
381     (#3)
382     + \int_div_truncate:nn { 26 * ( #2 + \c_one ) } \c_ten
383     + (#1)
384     + \int_div_truncate:nn {#1} \c_four
385     + \c_six * \int_div_truncate:nn {#1} \c_one_hundred
386     + \int_div_truncate:nn {#1} { 400 }
387   }
388   { \c_seven }
389 }

```

(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

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

```

390 \cs_new:Npn \zhnum_Zeller_Julian:nnn #1#2#3

```

<sup>1</sup>[http://en.wikipedia.org/wiki/Zeller's\\_congruence](http://en.wikipedia.org/wiki/Zeller's_congruence)

```

391 {
392   \int_mod:nn
393   {
394     (#3)
395     + \int_div_truncate:nn { 26 * ( #2 + \c_one ) } \c_ten
396     + (#1)
397     + \int_div_truncate:nn {#1} \c_four
398     + \c_five
399   }
400   { \c_seven }
401 }

```

(End definition for \zhnum\_Zeller\_Julian:nnn.)

**\zhptime** 输出时间。

```

402 \cs_new:Npn \zhptime #1
403 { \__zhnum_time:ww #1 \q_stop }
404 \group_begin:
405 \char_set_lccode:nn { \; } { \: }
406 \tl_to_lowercase:n
407 {
408   \group_end:
409   \cs_new:Npn \__zhnum_time:ww #1 ; #2 \q_stop
410   {
411     \zhnum_check_time:Nn \zhnum_int:n {#1} \c__zhnum_hour_tl
412     \zhnum_check_time:Nn \zhnum_int:n {#2} \c__zhnum_minute_tl
413   }
414 }

```

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

**\zhcurrtime** 输出当前时间。

```

415 \cs_new_nopar:Npn \zhcurrtime
416 {
417   \zhnum_check_time:Nn \zhnum_int:n
418   { \int_div_truncate:nn \tex_time:D { 60 } } \c__zhnum_hour_tl
419   \zhnum_check_time:Nn \zhnum_int:n
420   { \int_mod:nn \tex_time:D { 60 } } \c__zhnum_minute_tl
421 }

```

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

**\zhnum\_digit\_map:n** 阿拉伯数字与中文数字的映射。

```

422 \cs_new:Npn \zhnum_digit_map:n #1
423 { \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 424 \cs_new:Npn \zhnum_scale_map:n #1
425 {
426   \cs_if_exist_use:cF { c__zhnum_s #1 _tl }
427   { \zhnum_scale_map_hook:n {#1} }
428 }
429 \cs_new:Npn \zhnum_scale_map_loop:n #1
430 { \zhnum_scale_map:n { \int_mod:nn {#1} \l__zhnum_scale_int } }
431 \cs_generate_variant:Nn \zhnum_scale_map:n { f }
432 \int_new:N \l__zhnum_scale_int
433 \int_set_eq:NN \l__zhnum_scale_int \c_eleven
434 \cs_new_eq:NN \zhnum_scale_map_hook:n \zhnum_scale_map_loop:n
435 \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。

```

436 \NewDocumentCommand \zhnumExtendScaleMap { > { \TrimSpaces } +o +m }
437 {
438   \int_zero:N \l_tmpa_int
439   \clist_map_function:nN {#2} \zhnum_set_scale:n

```

```

440     \IfNoValueF {#1}
441     { \cs_set:Npn \zhnum_scale_map_hook:n ##1 {#1} }
442 }

```

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

\zhnum\_set\_scale:n

```

443 \cs_new_protected:Npn \zhnum_set_scale:n #1
444 {
445     \int_incr:N \l_tmpa_int
446     \tl_set:Nx \l_tmpa_tl
447     { c__zhnum_s \int_eval:n { \l_tmpa_int + \c_eleven } _tl }
448     \tl_if_exist:cF { \l_tmpa_tl }
449     { \int_incr:N \l__zhnum_scale_int }
450     \tl_set:cn { \l_tmpa_tl } {#1}
451 }

```

(End definition for \zhnum\_set\_scale:n.)

\zhnum\_ganzhi\_normal:nnn

保证干支的参数为正数。

```

452 \cs_new:Npn \zhnum_ganzhi_normal:nnn #1#2#3
453 {
454     \int_compare:nNnF {#1} < \c_one
455     { \cs_if_exist_use:c { c__zhnum_ #2 _ #1 _tl } }
456 }

```

(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

```

457 \cs_new:Npn \zhnum_ganzhi_cyclic:nnn #1#2#3
458 {
459     \int_compare:nNnF {#1} = \c_zero
460     {
461         \cs_if_exist_use:cF { c__zhnum_ #2 _ #1 _tl }
462         {
463             \__zhnum_ganzhi_cyclic_mod:fnnn
464             { \int_mod:nn {#1} {#3} } {#1} {#2} {#3}
465         }
466     }
467 }
468 \cs_new:Npn \__zhnum_ganzhi_cyclic_mod:nnnn #1#2#3#4
469 {
470     \int_compare:nNnTF {#2} > \c_zero
471     { \use:c { c__zhnum_ #3 _ #1 _tl } }
472     {
473         \int_compare:nNnTF {#1} = \c_zero
474         { \use:c { c__zhnum_ #3 _ 1 _tl } }
475         { \use:c { c__zhnum_ #3 _ \int_eval:n { #1 + #4 + 1 } _tl } }
476     }
477 }
478 \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。

```

479 \cs_new_eq:NN \zhnum_ganzhi:nnn \zhnum_ganzhi_normal:nnn
480 \cs_generate_variant:Nn \zhnum_ganzhi:nnn { f }

```

(End definition for \zhnum\_ganzhi:nnn.)

\zhtiangan 天干。

```

481 \cs_new:Npn \zhtiangan #1
482 { \zhnum_ganzhi:fnn { \int_eval:n {#1} } { tiangan } { 10 } }

```

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

\zhdizhi 地支。

```

483 \cs_new:Npn \zhdizhi #1
484 { \zhnum_ganzhi:fnn { \int_eval:n {#1} } { dizhi } { 12 } }

```

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

**\zhganzhi** 干支。

```
485 \cs_new:Npn \zhganzhi #1
486 { \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。

```
487 \cs_new:Npn \zhganzhinian #1
488 { \zhnum_ganzhi_nian:f { \int_eval:n {#1} } }
```

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

**\zhnum\_ganzhi\_nian:n** 干支纪年。公元元年是 \zhganzhi{58}。

```
489 \cs_new:Npn \zhnum_ganzhi_nian:n #1
490 {
491   \int_compare:nNnTF {#1} > \c_zero
492   { \use:c { c__zhnum_ganzhi_ \int_mod:nn { #1 + 57 } { 60 } _tl } }
493   {
494     \int_compare:nNnF {#1} = \c_zero
495     {
496       \use:c
497       {
498         c__zhnum_ganzhi_
499         \int_eval:n { \int_mod:nn { #1 - 2 } { 60 } + 60 }
500         _tl
501       }
502     }
503   }
504 }
505 \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。

```
506 \group_begin:
507 \tl_set:Nn \l_tmpa_tl
508 {
509   - .tl_set:N = \l__zhnum_minus_tl ,
510   -0 .tl_set:N = \l__zhnum_null_tl ,
511 }
512 \tl_put_right:Nx \l_tmpa_tl
513 {
514   E2 .tl_set:N = \exp_not:c { l__zhnum_ 100 _tl } ,
515   E3 .tl_set:N = \exp_not:c { l__zhnum_ 1000 _tl } ,
516   FE2 .tl_set:N = \exp_not:c { l__zhnum_financial_ 100 _tl } ,
517   FE3 .tl_set:N = \exp_not:c { l__zhnum_financial_ 1000 _tl } ,
518   D11 .tl_set:N = \exp_not:c { l__zhnum_dizhi_ 11 _tl } ,
519   D12 .tl_set:N = \exp_not:c { l__zhnum_dizhi_ 12 _tl } ,
520   E44 .tl_set:N = \exp_not:c { l__zhnum_ s11 _tl } ,
521 }
522 \int_step_inline:nnnn { 1 } { 1 } { 10 }
523 {
524   \tl_put_right:Nx \l_tmpa_tl
525   {
526     #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } ,
527     F#1 .tl_set:N = \exp_not:c { l__zhnum_financial_ #1 _tl } ,
528     T#1 .tl_set:N = \exp_not:c { l__zhnum_tiangang_ #1 _tl } ,
529     D#1 .tl_set:N = \exp_not:c { l__zhnum_dizhi_ #1 _tl } ,
530     GZ#1 .tl_set:N = \exp_not:c { l__zhnum_ganzhi_ #1 _tl } ,
531     E \int_eval:n { #1 * 4 }
532     .tl_set:N = \exp_not:c { l__zhnum_ s#1 _tl } ,
533   }
534 }
535 \int_step_inline:nnnn { 11 } { 1 } { 60 }
536 {
537   \tl_put_right:Nx \l_tmpa_tl
538   { GZ#1 .tl_set:N = \exp_not:c { l__zhnum_ganzhi_ #1 _tl } , }
```

```

539     }
540     \clist_map_inline:nn { 0 , 100 , 1000 }
541     {
542         \tl_put_right:Nx \l_tmpa_tl
543         {
544             #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } ,
545             F#1 .tl_set:N = \exp_not:c { l__zhnum_financial_ #1 _tl } ,
546         }
547     }
548     \clist_map_inline:nn { 20 , 30 , 40 , 200 }
549     {
550         \tl_put_right:Nx \l_tmpa_tl
551         { #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } , }
552     }
553     \clist_map_inline:nn
554     {
555         dot , and , parts , year , month , day , weekday , hour , minute
556         mon , tue , wed , thu , fri , sat , sun
557     }
558     {
559         \tl_put_right:Nx \l_tmpa_tl
560         { #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } , }
561     }
562     \use:x
563     {
564         \group_end:
565         \keys_define:nn { zhnum / options } { \exp_not:o \l_tmpa_tl }
566     }

```

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
567 \cs_new_protected:Npn \zhnum_set_digits_map:nn #1#2
568 { \prop_put:Nnn \l__zhnum_cfg_map_prop {#1} {#2} }
569 \cs_new_protected:Npn \zhnum_set_digits_map:nnn #1#2#3
570 {
571     \prop_put_if_new:Nnn \l__zhnum_cfg_map_prop {#1} {#3}
572     \prop_put:Nnn \l__zhnum_cfg_map_var_prop {#1_#2} {#3}
573 }
574 \cs_new_protected:Npn \zhnum_set_financial_map:nn #1#2
575 { \prop_put:Nnn \l__zhnum_cfg_map_finan_prop {#1} {#2} }
576 \cs_new_protected:Npn \zhnum_set_financial_map:nnn #1#2#3
577 {
578     \prop_put_if_new:Nnn \l__zhnum_cfg_map_finan_prop {#1} {#3}
579     \prop_put:Nnn \l__zhnum_cfg_map_var_prop { financial_#1_#2 } {#3}
580 }
581 \cs_new_protected:Npn \zhnum_set_tiangang_map:nn #1#2
582 { \prop_put:Nnn \l__zhnum_cfg_map_ganzhi_prop { tiangan_#1 } {#2} }
583 \cs_new_protected:Npn \zhnum_set_dizhi_map:nn #1#2
584 { \prop_put:Nnn \l__zhnum_cfg_map_ganzhi_prop { dizhi_#1 } {#2} }
585 \prop_new:N \l__zhnum_cfg_map_prop
586 \prop_new:N \l__zhnum_cfg_map_var_prop
587 \prop_new:N \l__zhnum_cfg_map_finan_prop
588 \prop_new:N \l__zhnum_cfg_map_ganzhi_prop

```

(End definition for \zhnum\_set\_digits\_map:nn and others.)

\zhnum\_parse\_config: 将 prop 表转化到单独的 tl 变量。

```

\zhnum_check_simp:nn
\zhnum_check_financial:nn
\zhnum_set_zero:
\zhnum_set_week_day:
589 \cs_new_protected_nopar:Npn \zhnum_parse_config:
590 {
591     \prop_map_function:NN \l__zhnum_cfg_map_prop \zhnum_check_simp:nn
592     \prop_map_function:NN \l__zhnum_cfg_map_ganzhi_prop \zhnum_assgin_ganzhi:nn
593     \zhnum_set_zero:
594     \zhnum_set_week_day:
595     \bool_if:NF \l__zhnum_reset_bool
596     {
597         \zhnum_assgin_const:
598         \bool_set_true:N \l__zhnum_reset_bool
599     }
600 }
601 \cs_new_protected:Npn \zhnum_check_simp:nn #1#2

```

```

602 {
603     \__zhnum_check_simp_aux:nn {#2} {#1}
604     \prop_get:NnNT \l__zhnum_cfg_map_finan_prop {#1} \l_tmpa_tl
605     { \exp_args:No \__zhnum_check_simp_aux:nn { \l_tmpa_tl } { financial_ #1 } }
606 }
607 \cs_new_protected:Npn \__zhnum_check_simp_aux:nn #1#2
608 {
609     \prop_get:NnNTF \l__zhnum_cfg_map_var_prop { #2 _trad } \l_tmpa_tl
610     {
611         \prop_get:NnNF \l__zhnum_cfg_map_var_prop { #2 _simp } \l_tmpb_tl
612         { \tl_set:Nn \l_tmpb_tl {#1} }
613         \tl_set:cx { l__zhnum_ #2 _tl }
614         {
615             \exp_not:n { \bool_if:NTF \l__zhnum_simp_bool }
616             { \exp_not:o \l_tmpb_tl } { \exp_not:o \l_tmpa_tl }
617         }
618     }
619     { \tl_set:cn { l__zhnum_ #2 _tl } {#1} }
620 }
621 \cs_new_protected_nopar:Npn \zhnum_assgin_const:
622 {
623     \prop_map_function:NN \l__zhnum_cfg_map_prop \zhnum_check_financial:nn
624     \zhnum_set_alias:
625 }
626 \cs_new_protected:Npn \zhnum_check_financial:nn #1#2
627 {
628     \prop_get:NnNTF \l__zhnum_cfg_map_finan_prop {#1} \l_tmpa_tl
629     {
630         \zhnum_assgin_const_tl:cx { c__zhnum_ #1 _tl }
631         {
632             \exp_not:n { \bool_if:NTF \l__zhnum_normal_bool }
633             { \exp_not:c { l__zhnum_ #1 _tl } }
634             { \exp_not:c { l__zhnum_financial_ #1 _tl } }
635         }
636     }
637     {
638         \zhnum_assgin_const_tl:cx
639         { c__zhnum_ #1 _tl } { \exp_not:c { l__zhnum_ #1 _tl } }
640     }
641 }
642 \cs_new_protected_nopar:Npn \zhnum_set_zero:
643 {
644     \tl_set:cx { l__zhnum_0_tl }
645     {
646         \exp_not:n { \bool_if:NTF \l__zhnum_null_bool }
647         { \exp_not:o \l__zhnum_null_tl } { \exp_not:v { l__zhnum_0_tl } }
648     }
649 }
650 \cs_new_protected_nopar:Npn \zhnum_set_week_day:
651 {
652     \tl_set:Nx \l__zhnum_mon_tl
653     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_1_tl } }
654     \tl_set:Nx \l__zhnum_tue_tl
655     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_2_tl } }
656     \tl_set:Nx \l__zhnum_wed_tl
657     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_3_tl } }
658     \tl_set:Nx \l__zhnum_thu_tl
659     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_4_tl } }
660     \tl_set:Nx \l__zhnum_fri_tl
661     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_5_tl } }
662     \tl_set:Nx \l__zhnum_sat_tl
663     { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_6_tl } }
664     \tl_set:Nx \l__zhnum_sun_tl
665     { \exp_not:N \c__zhnum_weekday_tl \exp_not:o \l__zhnum_day_tl }
666 }
667 \clist_map_inline:nn { mon , tue , wed , thu , fri , sat , sun }
668 { \tl_const:cx { c__zhnum_ #1 _tl } { \exp_not:c { l__zhnum_ #1 _tl } } }
669 \cs_new_protected:Npn \zhnum_assgin_ganzhi:nn #1#2
670 { \tl_set:cn { l__zhnum_ #1 _tl } {#2} }

```

```

671 \cs_new:Npn \zhnum_zero_mod:nn #1#2
672 { \exp_args:Nf \__zhnum_zero_mod_aux:nn { \int_mod:nn {#1} {#2} } {#2} }
673 \cs_new:Npn \__zhnum_zero_mod_aux:nn #1#2
674 { \int_compare:nNnTF {#1} = \c_zero {#2} {#1} }
675 \int_step_inline:nnnn { 1 } { 1 } { 60 }
676 {
677   \tl_const:cx { c__zhnum_ganzhi_ #1 _tl } { \exp_not:c { l__zhnum_ganzhi_ #1 _tl } }
678   \tl_set:cx { l__zhnum_ganzhi_ #1 _tl }
679   {
680     \exp_not:c { l__zhnum_tiangang_ \zhnum_zero_mod:nn {#1} { 10 } _tl }
681     \exp_not:c { l__zhnum_dizhi_ \zhnum_zero_mod:nn {#1} { 12 } _tl }
682   }
683 }
684 \cs_new_eq:cc { c__zhnum_ganzhi_ 0 _tl } { c__zhnum_ganzhi_ 60 _tl }
685 \cs_new_eq:NN \zhnum_assgin_const_tl:cx \tl_const:cx
686 \AtEndOfPackage
687 {
688   \prop_map_inline:Nn \l__zhnum_cfg_map_ganzhi_prop
689   { \tl_const:cx { c__zhnum_ #1 _tl } { \exp_not:c { l__zhnum_ #1 _tl } } }
690   \cs_new_eq:cc { c__zhnum_tiangang_ 0 _tl } { c__zhnum_tiangang_ 10 _tl }
691   \cs_new_eq:cc { c__zhnum_dizhi_ 0 _tl } { c__zhnum_dizhi_ 12 _tl }
692   \cs_set_eq:NN \zhnum_assgin_const_tl:cx \tl_set:cx
693 }

```

(End definition for \zhnum\_parse\_config: and others.)

\zhnum\_set\_alias: 一些易于使用的别名。

```

694 \cs_new_eq:NN \zhnum_set_alias:NN \cs_new_eq:NN
695 \cs_new_protected_nopar:Npx \zhnum_set_alias:
696 {
697   \zhnum_set_alias:NN \exp_not:N \c__zhnum_zero_tl
698   \exp_not:c { c__zhnum_ 0 _tl }
699   \zhnum_set_alias:NN \exp_not:N \c__zhnum_ten_tl
700   \exp_not:c { c__zhnum_ 10 _tl }
701   \zhnum_set_alias:NN \exp_not:N \c__zhnum_hundred_tl
702   \exp_not:c { c__zhnum_ 100 _tl }
703   \zhnum_set_alias:NN \exp_not:N \c__zhnum_thousand_tl
704   \exp_not:c { c__zhnum_ 1000 _tl }
705 }
706 \AtEndOfPackage
707 { \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。

```

708 \cs_new_protected:Npn \zhnum_load_cfg:n #1
709 {
710   \zhnum_set_cfg_name:Nn \l__zhnum_cfg_tl {#1}
711   \tl_if_eq:NNF \l__zhnum_cfg_tl \l__zhnum_last_cfg_tl
712   { \zhnum_update_cfg:n {#1} }
713   \zhnum_parse_config:
714 }
715 \cs_generate_variant:Nn \zhnum_load_cfg:n { o }
716 \cs_new_protected:Npn \zhnum_update_cfg:n #1
717 {
718   \prop_if_exist:cTF { g__zhnum_cfg_ \l__zhnum_cfg_tl _prop }
719   { \tl_set_eq:NN \l__zhnum_last_cfg_tl \l__zhnum_cfg_tl }
720   { \zhnum_input_cfg:n {#1} }
721   \__zhnum_update_cfg_prop:N \prop_set_eq:Nc
722 }
723 \cs_new_protected:Npn \zhnum_input_cfg:n #1
724 {
725   \file_if_exist_input:nTF { zhnumber - #1 .cfg }
726   {
727     \bool_set_false:N \l__zhnum_reset_bool
728     \__zhnum_update_cfg_prop:N \__zhnum_prop_initial:Nn
729     \group_begin:
730     \zhnum_set_catcode:
731   }

```

```

732     {
733         \msg_error:nxx { zhnumber } { file-not-found } {#1}
734         \use_none:nnn
735     }
736     \__zhnum_update_cfg_prop:N \__zhnum_prop_gset_eq:Nn
737     \group_end:
738 }
739 \cs_new_protected:Npn \__zhnum_update_cfg_prop:N #1
740 {
741     #1 \l__zhnum_cfg_map_prop      { g__zhnum_cfg_ \l__zhnum_cfg_tl _prop }
742     #1 \l__zhnum_cfg_map_var_prop  { g__zhnum_cfg_var_ \l__zhnum_cfg_tl _prop }
743     #1 \l__zhnum_cfg_map_finan_prop { g__zhnum_cfg_finan_ \l__zhnum_cfg_tl _prop }
744     #1 \l__zhnum_cfg_map_ganzhi_prop { g__zhnum_cfg_ganzhi_ \l__zhnum_cfg_tl _prop }
745 }
746 \cs_new_protected:Npn \__zhnum_prop_initial:Nn #1#2
747 {
748     \prop_clear:N #1
749     \prop_new:c {#2}
750 }
751 \cs_new_protected:Npn \__zhnum_prop_gset_eq:Nn #1#2
752 { \prop_gset_eq:cN {#2} #1 }
753 \tl_new:N \l__zhnum_cfg_tl
754 \tl_new:N \l__zhnum_last_cfg_tl
755 \bool_new:N \l__zhnum_reset_bool
756 \msg_new:nnnn { zhnumber } { file-not-found }
757 { File~`#1'~not~found. }
758 {
759     The~requested~file~could~not~be~found~in~the~current~directory,~
760     in~the~TeX~search~path~or~in~the~LaTeX~search~path.
761 }

```

(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。因此, 参考 `ifuptex` 宏包, 通过 `\kchar` 是否为 `primitive` 来判断。

```

762 \pdfTeX_if_engine:TF
763 {
764     \str_if_eq_x:nnTF
765         { \token_to_str:N \kchar }
766         { \token_to_meaning:N \kchar }
767 }
768 { \use_i:nn }
769 {
770     \cs_new_eq:NN \zhnum_if_unicode_engine_p: \c_true_bool
771     \cs_new_eq:NN \zhnum_if_unicode_engine:TF \use_i:nn
772 }
773 {
774     \cs_new_eq:NN \zhnum_if_unicode_engine_p: \c_false_bool
775     \cs_new_eq:NN \zhnum_if_unicode_engine:TF \use_ii:nn
776 }

```

(End definition for `\zhnum_if_unicode_engine:TF`.)

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

```

\zhnum_set_cfg_name:Nn 777 \if_predicate:w \zhnum_if_unicode_engine_p:
\zhnum_reset_config: 778 \cs_new_eq:NN \zhnum_set_catcode: \prg_do_nothing:
779 \cs_new_protected:Npn \zhnum_set_cfg_name:Nn #1#2
780 {
781     \tl_set:Nx \l__zhnum_encoding_tl {#2}
782     \tl_set:Nx #1 { \tl_to_str:N \l__zhnum_encoding_tl }
783 }
784 \cs_new_eq:NN \zhnum_reset_config: \zhnum_parse_config:
785 \else:
786 \cs_new_protected_nopar:Npn \zhnum_set_catcode:
787 { \bool_if:NT \l__zhnum_active_char_bool { \zhnum_set_active: } }
788 \cs_new_protected_nopar:Npn \zhnum_set_active:
789 {
790     \str_case:onTF { \l__zhnum_encoding_tl }
791     {

```

```

792         { gbk } { \int_set:Nn \l__zhnum_byte_min_int { "81 } }
793         { big5 } { \int_set:Nn \l__zhnum_byte_min_int { "A1 } }
794     }
795     { \int_set:Nn \l__zhnum_byte_max_int { "FE } }
796     {
797         \int_set:Nn \l__zhnum_byte_min_int { "E0 }
798         \int_set:Nn \l__zhnum_byte_max_int { "EF }
799     }
800     \int_step_function:nnnN
801     { \l__zhnum_byte_min_int } { \c_one }
802     { \l__zhnum_byte_max_int } \char_set_catcode_active:n
803 }
804 \int_new:N \l__zhnum_byte_min_int
805 \int_new:N \l__zhnum_byte_max_int
806 \cs_new_protected:Npn \zhnum_set_cfg_name:Nn #1#2
807 {
808     \tl_set:Nx \l__zhnum_encoding_tl {#2}
809     \tl_set:Nx #1
810     {
811         \tl_to_str:N \l__zhnum_encoding_tl
812         \bool_if:NT \l__zhnum_active_char_bool
813         { \tl_to_str:n { _active } }
814     }
815 }
816 \cs_new_protected_nopar:Npn \zhnum_reset_config:
817 { \zhnum_load_cfg:o { \l__zhnum_encoding_tl } }
818 \bool_new:N \l__zhnum_active_char_bool
819 \bool_set_true:N \l__zhnum_active_char_bool
820 \fi:

```

(End definition for \zhnum\_set\_catcode:, \zhnum\_set\_cfg\_name:Nn, and \zhnum\_reset\_config:.)

encoding 宏包设置选项。

```

style 821 \keys_define:nn { zhnum / options }
null 822 {
reset 823     encoding .choices:nn =
824     { UTF8 , GBK , Big5 }
825     {
826         \exp_args:Nx \tex_lowercase:D
827         { \tl_set:Nn \exp_not:N \l__zhnum_encoding_tl { \l_keys_choice_tl } }
828         \zhnum_load_cfg:o { \l__zhnum_encoding_tl }
829     } ,
830     encoding .default:n = { GBK } ,
831     encoding / Bg5 .meta:n = { encoding = Big5 } ,
832     encoding / unknown .code:n =
833     { \msg_error:nnn { zhnumber } { encoding-invalid } {#1} } ,
834     style .multichoice: ,
835     style / Normal .code:n =
836     {
837         \bool_set_false:N \l__zhnum_ancient_bool
838         \bool_set_true:N \l__zhnum_normal_bool
839     } ,
840     style / Financial .code:n =
841     {
842         \bool_set_false:N \l__zhnum_ancient_bool
843         \bool_set_false:N \l__zhnum_normal_bool
844     } ,
845     style / Ancient .code:n =
846     {
847         \bool_set_true:N \l__zhnum_ancient_bool
848         \bool_set_true:N \l__zhnum_normal_bool
849     } ,
850     style / Simplified .code:n = { \bool_set_true:N \l__zhnum_simp_bool } ,
851     style / Traditional .code:n = { \bool_set_false:N \l__zhnum_simp_bool } ,
852     style .default:n = { Normal , Simplified } ,
853     null .bool_set:N = \l__zhnum_null_bool ,
854     time .choice: ,
855     time / Chinese .code:n = { \bool_set_true:N \l__zhnum_time_bool } ,
856     time / Arabic .code:n = { \bool_set_false:N \l__zhnum_time_bool } ,

```

```

857     time                .default:n = { Arabic } ,
858     reset                .code:n = { \zhnum_reset_config: } ,
859     activechar           .bool_set:N = \l__zhnum_active_char_bool ,
860     ganzhi-cyclic .choice: ,
861     ganzhi-cyclic / true .code:n =
862     { \cs_set_eq:NN \zhnum_ganzhi:nnn \zhnum_ganzhi_cyclic:nnn } ,
863     ganzhi-cyclic / false.code:n =
864     { \cs_set_eq:NN \zhnum_ganzhi:nnn \zhnum_ganzhi_normal:nnn } ,
865     ganzhi-cyclic       .default:n = { true } ,
866   }
867   \tl_new:N \l__zhnum_encoding_tl
868   \msg_new:nnnn { zhnumber } { encoding-invalid }
869   { The~encoding~`#1'~is~invalid. }
870   { 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。

```

871 \NewDocumentCommand \zhnumsetup { +m }
872 {
873   \keys_set:nn { zhnum / options } {#1}
874   \tex_ignorespaces:D
875 }

```

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

初始化设置和执行宏包选项。

```

876 \keys_set:nn { zhnum / options } { style , time }
877 \ProcessKeysOptions { zhnum / options }
878 如果没有选定编码,则根据引擎自动设置编码。
879 \tl_if_empty:NT \l__zhnum_encoding_tl
880 {
881   \zhnum_if_unicode_engine:TF
882   { \keys_set:nn { zhnum / options } { encoding = UTF8 } }
883   { \keys_set:nn { zhnum / options } { encoding = GBK } }
884 }
885 \package

```

## 4 中文数字配置文件

```

885 <*config>
886 <!*big5>
887 \zhnum_set_digits_map:nnn { minus } { simp } { 负 }
888 \zhnum_set_digits_map:nnn { minus } { trad } { 負 }
889 </!big5>
890 <*big5>
891 \zhnum_set_digits_map:nn { minus } { 負 }
892 </big5>
893 \zhnum_set_digits_map:nn { 0 } { 零 }
894 <!*big5>
895 \zhnum_set_digits_map:nn { null } { 〇 }
896 </!big5>
897 <*big5>
898 \zhnum_set_digits_map:nn { null } { 〇 }
899 </big5>
900 \zhnum_set_digits_map:nn { 1 } { 一 }
901 \zhnum_set_digits_map:nn { 2 } { 二 }
902 \zhnum_set_digits_map:nn { 3 } { 三 }
903 \zhnum_set_digits_map:nn { 4 } { 四 }
904 \zhnum_set_digits_map:nn { 5 } { 五 }
905 \zhnum_set_digits_map:nn { 6 } { 六 }
906 \zhnum_set_digits_map:nn { 7 } { 七 }
907 \zhnum_set_digits_map:nn { 8 } { 八 }
908 \zhnum_set_digits_map:nn { 9 } { 九 }
909 \zhnum_set_digits_map:nn { 10 } { 十 }
910 \zhnum_set_digits_map:nn { 100 } { 百 }
911 \zhnum_set_digits_map:nn { 1000 } { 千 }

```

```

912 \zhnum_set_digits_map:nn { 20 } { 廿 }
913 \zhnum_set_digits_map:nn { 30 } { 卅 }
914 \zhnum_set_digits_map:nn { 40 } { 卌 }
915 \zhnum_set_digits_map:nn { 200 } { 兩 }
916  $\langle$ !big5 $\rangle$ 
917 \zhnum_set_digits_map:nnn { dot } { simp } { 点 }
918 \zhnum_set_digits_map:nnn { dot } { trad } { 點 }
919  $\langle$ /!big5 $\rangle$ 
920  $\langle$ *big5 $\rangle$ 
921 \zhnum_set_digits_map:nn { dot } { 點 }
922  $\langle$ /big5 $\rangle$ 
923 \zhnum_set_digits_map:nn { and } { 又 }
924 \zhnum_set_digits_map:nn { parts } { 分之 }
925  $\langle$ !big5 $\rangle$ 
926 \zhnum_set_digits_map:nnn { s1 } { simp } { 万 }
927 \zhnum_set_digits_map:nnn { s1 } { trad } { 萬 }
928 \zhnum_set_digits_map:nnn { s2 } { simp } { 亿 }
929 \zhnum_set_digits_map:nnn { s2 } { trad } { 億 }
930  $\langle$ /!big5 $\rangle$ 
931  $\langle$ *big5 $\rangle$ 
932 \zhnum_set_digits_map:nn { s1 } { 萬 }
933 \zhnum_set_digits_map:nn { s2 } { 億 }
934  $\langle$ /big5 $\rangle$ 
935 \zhnum_set_digits_map:nn { s3 } { 兆 }
936 \zhnum_set_digits_map:nn { s4 } { 京 }
937 \zhnum_set_digits_map:nn { s5 } { 垓 }
938 \zhnum_set_digits_map:nn { s6 } { 秭 }
939 \zhnum_set_digits_map:nn { s7 } { 穰 }
940  $\langle$ !big5 $\rangle$ 
941 \zhnum_set_digits_map:nnn { s8 } { simp } { 沟 }
942 \zhnum_set_digits_map:nnn { s8 } { trad } { 溝 }
943 \zhnum_set_digits_map:nnn { s9 } { simp } { 涧 }
944 \zhnum_set_digits_map:nnn { s9 } { trad } { 澗 }
945  $\langle$ /!big5 $\rangle$ 
946  $\langle$ *big5 $\rangle$ 
947 \zhnum_set_digits_map:nn { s8 } { 溝 }
948 \zhnum_set_digits_map:nn { s9 } { 澗 }
949  $\langle$ /big5 $\rangle$ 
950 \zhnum_set_digits_map:nn { s10 } { 正 }
951  $\langle$ !big5 $\rangle$ 
952 \zhnum_set_digits_map:nnn { s11 } { simp } { 载 }
953 \zhnum_set_digits_map:nnn { s11 } { trad } { 載 }
954  $\langle$ /!big5 $\rangle$ 
955  $\langle$ *big5 $\rangle$ 
956 \zhnum_set_digits_map:nn { s11 } { 載 }
957  $\langle$ /big5 $\rangle$ 
958 \zhnum_set_digits_map:nn { year } { 年 }
959 \zhnum_set_digits_map:nn { month } { 月 }
960 \zhnum_set_digits_map:nn { day } { 日 }
961  $\langle$ !big5 $\rangle$ 
962 \zhnum_set_digits_map:nnn { hour } { simp } { 时 }
963 \zhnum_set_digits_map:nnn { hour } { trad } { 時 }
964  $\langle$ /!big5 $\rangle$ 
965  $\langle$ *big5 $\rangle$ 
966 \zhnum_set_digits_map:nn { hour } { 時 }
967  $\langle$ /big5 $\rangle$ 
968 \zhnum_set_digits_map:nn { minute } { 分 }
969 \zhnum_set_digits_map:nn { weekday } { 星期 }
970 \zhnum_set_financial_map:nn { null } { 零 }
971 \zhnum_set_financial_map:nn { 0 } { 零 }
972 \zhnum_set_financial_map:nn { 1 } { 壹 }
973 \zhnum_set_financial_map:nn { 2 } { 貳 }
974  $\langle$ !big5 $\rangle$ 
975 \zhnum_set_financial_map:nnn { 3 } { simp } { 叁 }
976 \zhnum_set_financial_map:nnn { 3 } { trad } { 參 }
977  $\langle$ /!big5 $\rangle$ 
978  $\langle$ *big5 $\rangle$ 
979 \zhnum_set_financial_map:nn { 3 } { 參 }
980  $\langle$ /big5 $\rangle$ 

```

```

981 \zhnum_set_financial_map:nn { 4 } { 肆 }
982 \zhnum_set_financial_map:nn { 5 } { 伍 }
983 <*\big5>
984 \zhnum_set_financial_map:nnn { 6 } { simp } { 陆 }
985 \zhnum_set_financial_map:nnn { 6 } { trad } { 陸 }
986 </\big5>
987 <*\big5>
988 \zhnum_set_financial_map:nn { 6 } { 陸 }
989 </\big5>
990 \zhnum_set_financial_map:nn { 7 } { 柒 }
991 \zhnum_set_financial_map:nn { 8 } { 捌 }
992 \zhnum_set_financial_map:nn { 9 } { 玖 }
993 \zhnum_set_financial_map:nn { 10 } { 拾 }
994 \zhnum_set_financial_map:nn { 100 } { 佰 }
995 \zhnum_set_financial_map:nn { 1000 } { 仟 }
996 \zhnum_set_tiangang_map:nn { 1 } { 甲 }
997 \zhnum_set_tiangang_map:nn { 2 } { 乙 }
998 \zhnum_set_tiangang_map:nn { 3 } { 丙 }
999 \zhnum_set_tiangang_map:nn { 4 } { 丁 }
1000 \zhnum_set_tiangang_map:nn { 5 } { 戊 }
1001 \zhnum_set_tiangang_map:nn { 6 } { 己 }
1002 \zhnum_set_tiangang_map:nn { 7 } { 庚 }
1003 \zhnum_set_tiangang_map:nn { 8 } { 辛 }
1004 \zhnum_set_tiangang_map:nn { 9 } { 壬 }
1005 \zhnum_set_tiangang_map:nn { 10 } { 癸 }
1006 \zhnum_set_dizhi_map:nn { 1 } { 子 }
1007 \zhnum_set_dizhi_map:nn { 2 } { 丑 }
1008 \zhnum_set_dizhi_map:nn { 3 } { 寅 }
1009 \zhnum_set_dizhi_map:nn { 4 } { 卯 }
1010 \zhnum_set_dizhi_map:nn { 5 } { 辰 }
1011 \zhnum_set_dizhi_map:nn { 6 } { 巳 }
1012 \zhnum_set_dizhi_map:nn { 7 } { 午 }
1013 \zhnum_set_dizhi_map:nn { 8 } { 未 }
1014 \zhnum_set_dizhi_map:nn { 9 } { 申 }
1015 \zhnum_set_dizhi_map:nn { 10 } { 酉 }
1016 \zhnum_set_dizhi_map:nn { 11 } { 戌 }
1017 \zhnum_set_dizhi_map:nn { 12 } { 亥 }
1018 </config>

```

## 5 代码索引

斜体的数字表示对应项说明所在的页码,下划线的数字表示定义所在的代码行号,而直立的数字表示对应项使用时所在的行号。

Symbols		E
\:	405	eleven commands:
\;	405	\c_eleven ..... 433, 447
\\	5, 6, 7	else commands:
		\else: ..... 124, 136, 167, 289, 318, 785
A		encoding ..... 1, 19
activechar	4	exp commands:
\AtEndOfPackage	686, 706	\exp_after:wN ..... 112, 114, 122, 123, 125,
		127, 128, 134, 135, 137, 138, 146, 166, 168, 188, 191,
		192, 193, 278, 280, 287, 288, 290, 291, 292, 293, 302, 304
B		\exp_args:Nf ..... 172, 174, 672
bingint commands:		\exp_args:No ..... 605
\__bingint_read_do:nn	6	\exp_args:Nx ..... 826
bool commands:		\exp_not:c ..... 514, 515, 516, 517, 518, 519, 520,
\bool_if:NF	211, 224, 595	526, 527, 528, 529, 530, 532, 538, 544, 545, 551, 560,
\bool_if:nF	237	633, 634, 639, 668, 677, 680, 681, 689, 698, 700, 702, 704
\bool_if:NT	787, 812	\exp_not:N ..... 121, 133, 165, 286, 302,
\bool_if:nTF	343, 615, 632, 646	653, 655, 657, 659, 661, 663, 665, 697, 699, 701, 703, 827
\bool_new:N	229, 243	\exp_not:n ..... 615, 632, 646
\bool_new:N	755, 818	\exp_not:o ..... 565, 616, 647, 665
\bool_set_false:N	727, 837, 842, 843, 851, 856	\exp_not:v ..... 647, 653, 655, 657, 659, 661, 663
\bool_set_true:N	598, 819, 838, 847, 848, 850, 855	\exp_stop_f: ..... 116, 165, 190, 195, 282, 348
\BooleanFalse	274	
\BooleanTrue	272	F
		false commands:
		\c_false_bool ..... 216, 217, 774
C		fi commands:
char commands:		\fi: ... 121, 129, 139, 169, 194, 286, 294, 302, 320, 356, 820
\char_set_catcode_active:n	802	file commands:
\char_set_lccode:nn	405	\file_if_exist_input:nTF ..... 725
clist commands:		five commands:
\clist_map_function:nN	439	\c_five ..... 398
\clist_map_inline:nn	540, 548, 553, 667	four commands:
cs commands:		\c_four ..... 174, 181, 189, 248, 384, 397
\cs:w	314	
\cs_end:	322	G
\cs_generate_variant:Nn		ganzhi-cyclic ..... 3
..... 35, 109, 185, 204, 220, 275, 431, 478, 480, 505, 715		group commands:
\cs_if_exist_use:c	455	\group_begin: ..... 22, 83, 266, 404, 506, 729
\cs_if_exist_use:cF	426, 461	\group_end: ..... 25, 86, 269, 408, 564, 737
\cs_new:Npn	27, 29, 36, 42, 61, 68, 88, 94,	
96, 110, 119, 131, 141, 149, 151, 153, 163, 171, 173, 175,		I
186, 197, 199, 205, 221, 276, 284, 296, 307, 312, 330,		if commands:
342, 344, 346, 358, 366, 372, 377, 390, 402, 409, 422,		\if:w ..... 121, 133, 286, 302
424, 429, 452, 457, 468, 481, 483, 485, 487, 489, 671, 673		\if_case:w ..... 190, 348
\cs_new_eq:cc	684, 690, 691	\if_int_compare:w ..... 165, 316
\cs_new_eq:NN	434, 479, 685, 694, 770, 771, 774, 775, 778, 784	\if_predicate:w ..... 777
\cs_new_nopar:Npn	271, 273, 336, 415	\IfBooleanT ..... 327
\cs_new_protected:Npn		\IfBooleanTF ..... 317
..... 443, 567, 569, 574, 576, 581, 583,		\IfNoValueF ..... 440
601, 607, 626, 669, 708, 716, 723, 739, 746, 751, 779, 806		\IfNoValueTF ..... 15, 76, 259
\cs_new_protected_nopar:Npn		int commands:
..... 589, 621, 642, 650, 786, 788, 816		\int_compare:nNnF . 158, 227, 235, 255, 309, 454, 459, 494
\cs_new_protected_nopar:Npx	695	\int_compare:nNnT ..... 374
\cs_set:Npn	441	\int_compare:nNnTF ..... 98, 101, 155, 177,
\cs_set_eq:NN	692, 707, 862, 864	180, 208, 214, 223, 226, 234, 360, 368, 470, 473, 491, 674
		\int_compare_p:nNn ..... 230, 239, 240, 246, 247, 248
		\int_div_truncate:nn 189, 382, 384, 385, 386, 395, 397, 418
D		
\DeclareExpandableDocumentCommand	13, 74, 257, 324	

\int_eval:n	99, 104, 152, 181, 218, 375, 447, 475, 482, 484, 486, 488, 499, 531
\int_if_exist:cTF	90
\int_incr:N	445, 449
\int_mod:nn	174, 379, 392, 420, 430, 464, 492, 499, 672
\int_new:N	432, 804, 805
\int_set:Nn	792, 793, 795, 797, 798
\int_set_eq:NN	433
\int_step_function:nnnN	800
\int_step_inline:nnnn	522, 535, 675
\int_to_arabic:n	343
\int_zero:N	438
<b>K</b>	
\kchar	765, 766
keys commands:	
\l_keys_choice_tl	827
\keys_define:nn	565, 821
\keys_set:nn	23, 84, 267, 873, 876, 881, 882
<b>M</b>	
mark commands:	
\q_mark	40, 42
msg commands:	
\msg_error:nn	11
\msg_error:nnn	833
\msg_error:nnx	733
\__msg_expandable_error:n	95
\msg_new:nnn	3
\msg_new:nnnn	756, 868
<b>N</b>	
\NewDocumentCommand	20, 81, 264, 436, 871
nil commands:	
\q_nil	8, 28, 32, 40, 202
nine commands:	
\c_nine	165
null	3, 19
<b>O</b>	
one commands:	
\c_one	144, 158, 181, 218, 239, 309, 369, 382, 395, 454, 801
\c_one_hundred	385
or commands:	
\or:	191, 192, 193, 350, 351, 352, 353, 354, 355
<b>P</b>	
pdftex commands:	
\pdfTeX_if_engine:TF	762
prg commands:	
\prg_do_nothing:	778
\ProcessKeysOptions	877
prop commands:	
\prop_clear:N	748
\prop_get:NnNF	611
\prop_get:NnNT	604
\prop_get:NnNTF	609, 628
\prop_gset_eq:cN	752
\prop_if_exist:cTF	718
\prop_map_function:NN	591, 592, 623
\prop_map_inline:Nn	688
\prop_new:c	749
\prop_new:N	585, 586, 587, 588
\prop_put:Nnn	568, 572, 575, 579, 582, 584
\prop_put_if_new:Nnn	571, 578
\prop_set_eq:Nc	721
<b>Q</b>	
quark commands:	
\quark_if_nil:nTF	31, 38, 44
\quark_if_recursion_tail_stop:N	207, 301
\quark_if_recursion_tail_stop_do:Nn	145
<b>R</b>	
recursion commands:	
\q_recursion_stop	116, 141, 147, 202, 282
\q_recursion_tail	8, 116, 202, 282
\RequirePackage	12
reset	4, 19
<b>S</b>	
seven commands:	
\c_seven	388, 400
six commands:	
\c_six	385
stop commands:	
\q_stop	28, 29, 32, 36, 40, 42, 326, 328, 330, 345, 346, 403, 409
str commands:	
\str_case:onTF	790
\str_if_eq_x:nnTF	764
style	3, 19
<b>T</b>	
ten commands:	
\c_ten	374, 382, 395
T <sub>E</sub> X and L <sup>A</sup> T <sub>E</sub> X 2 <sub>ε</sub> commands:	
\ifpackagelater	10
\kchar	18
\tiangan	3
\zhdate	2, 2
\zhdigits	1, 1, 1, 1, 2, 3, 4
\zhdizhi	2
\zhganzhi	2
\zhganzhinian	2
\zhnum	1, 1, 4
\zhnumber	1, 1, 2, 2, 4
\zhnumExtendScaleMap	2, 2
\zhnumsetup	1, 3, 4
\zhtiangan	2
\zhtime	2
\zhweekday	2
tex commands:	
\tex_day:D	340
\tex_ignorespaces:D	874
\tex_lowercase:D	826
\tex_month:D	339
\tex_number:D	113, 189, 279
\tex_romannumeral:D	115, 123, 126, 135, 147, 281, 288, 305
\tex_time:D	418, 420
\tex_year:D	338
three commands:	
\c_three	247, 368
time	3
tl commands:	
\__tl_act:NNNnn	7
\tl_const:cn	435
\tl_const:cx	668, 677, 685, 689

\tl\_count:n ..... 172  
 \tl\_if\_blank:nF ..... 51  
 \tl\_if\_blank:nTF ..... 64, 70  
 \tl\_if\_empty:NT ..... 878  
 \tl\_if\_eq:NNF ..... 711  
 \tl\_if\_exist:cF ..... 448  
 \tl\_new:N ..... 753, 754, 867  
 \tl\_put\_right:Nx ..... 512, 524, 537, 542, 550, 559  
 \tl\_set:cn ..... 450, 619, 670  
 \tl\_set:cx ..... 613, 644, 678, 692  
 \tl\_set:Nn ..... 507, 612, 827  
 \tl\_set:Nx .....  
 . 446, 652, 654, 656, 658, 660, 662, 664, 781, 782, 808, 809  
 \tl\_set\_eq:NN ..... 707, 719  
 \tl\_to\_lowercase:n ..... 406  
 \tl\_to\_str:N ..... 782, 811  
 \tl\_to\_str:n ..... 813  
 tmpa commands:  
 \l\_tmpa\_int ..... 438, 445, 447  
 \l\_tmpa\_tl ..... 446, 448, 450, 507,  
 512, 524, 537, 542, 550, 559, 565, 604, 605, 609, 616, 628  
 tmpb commands:  
 \l\_tmpb\_tl ..... 611, 612, 616  
 token commands:  
 \token\_to\_meaning:N ..... 766  
 \token\_to\_str:N ..... 765  
 \TrimSpaces ..... 436  
 true commands:  
 \c\_true\_bool ..... 201, 216, 217, 770  
 twelve commands:  
 \c\_twelve ..... 369  
 two commands:  
 \c\_two ..... 177, 230, 246

## U

use commands:  
 \use:c ..... 423, 471, 474, 475, 492, 496  
 \use:n ..... 115, 123, 135, 147, 191, 281, 288, 305  
 \use:x ..... 562  
 \use\_i:nn ..... 166, 209, 768, 771  
 \use\_i:nnn ..... 193  
 \use\_i\_ii:nnn ..... 192  
 \use\_ii:nn ..... 168, 775  
 \use\_none:n ..... 250  
 \use\_none:nnn ..... 734

## Z

zero commands:  
 \c\_zero .... 98, 101, 117, 155, 180, 208, 214, 223, 226,  
 227, 234, 235, 240, 255, 316, 459, 470, 473, 491, 494, 674  
 \zhcurrtime ..... 2, 12, 415  
 \zhdate ..... 2, 10, 324  
 \zhdigits ..... 1, 4, 9, 257  
 \zhdigitsoptions ..... 9, 261, 264  
 \zh dizhi ..... 2, 13, 483  
 \zh ganzhi ..... 2, 14, 485  
 \zh ganzhinian ..... 2, 14, 487  
 \zhnum ..... 1, 4, 5, 74  
 zhnum commands:  
 \l\_zhnum\_active\_char\_bool .... 787, 812, 818, 819, 859  
 \l\_zhnum\_ancient\_bool ..... 230, 245, 837, 842, 847  
 \c\_zhnum\_and\_tl ..... 54  
 \zhnum\_assgin\_const: ..... 597, 621

\zhnum\_assgin\_const\_tl:cx ..... 630, 638, 685, 692  
 \zhnum\_assgin\_ganzhi:nn ..... 592, 669  
 \zhnum\_blank\_to\_zero:n ..... 5, 46, 48, 56, 58, 63, 68  
 \l\_zhnum\_byte\_max\_int ..... 795, 798, 802, 805  
 \l\_zhnum\_byte\_min\_int ..... 792, 793, 797, 801, 804  
 \l\_zhnum\_cfg\_map\_finan\_prop .....  
 ..... 15, 575, 578, 587, 604, 628, 743  
 \l\_zhnum\_cfg\_map\_ganzhi\_prop .....  
 ..... 15, 582, 584, 588, 592, 688, 744  
 \l\_zhnum\_cfg\_map\_prop . 15, 568, 571, 585, 591, 623, 741  
 \l\_zhnum\_cfg\_map\_var\_prop .....  
 ..... 15, 572, 579, 586, 609, 611, 742  
 \l\_zhnum\_cfg\_tl 710, 711, 718, 719, 741, 742, 743, 744, 753  
 \zhnum\_check\_financial:nn ..... 15, 623, 626  
 \zhnum\_check\_simp:nn ..... 15, 591, 601  
 \\_zhnum\_check\_simp\_aux:nn ..... 603, 605, 607  
 \zhnum\_check\_time:Nn .....  
 . . 10, 332, 333, 334, 338, 339, 340, 342, 411, 412, 417, 419  
 \zhnum\_counter:n ..... 6, 77, 85, 88  
 \\_zhnum\_counter\_error:n ..... 92, 94  
 \\_zhnum\_date:www ..... 326, 330  
 \c\_zhnum\_day\_tl ..... 334, 340  
 \l\_zhnum\_day\_tl ..... 665  
 \zhnum\_decimal:nn ..... 5, 33, 61  
 \zhnum\_digit\_map:n .....  
 ..... 12, 178, 225, 231, 232, 250, 251, 255, 422  
 \zhnum\_digits:Nn ..... 9, 260, 268, 272, 274, 276  
 \zhnum\_digits\_null:n ..... 9, 273, 275, 332  
 \zhnum\_digits\_null:V ..... 338  
 \zhnum\_digits\_zero:n ..... 9, 66, 271  
 \c\_zhnum\_dot\_tl ..... 63, 302  
 \l\_zhnum\_encoding\_tl .....  
 ..... 781, 782, 790, 808, 811, 817, 827, 828, 867, 878  
 \\_zhnum\_fraction:www ..... 5, 40, 42  
 \c\_zhnum\_fri\_tl ..... 355  
 \l\_zhnum\_fri\_tl ..... 660  
 \zhnum\_ganzhi:fnn ..... 482, 484, 486  
 \zhnum\_ganzhi:nnn ..... 13, 479, 480, 862, 864  
 \zhnum\_ganzhi\_cyclic:nnn ..... 13, 457, 862  
 \\_zhnum\_ganzhi\_cyclic\_mod:fnnn ..... 463  
 \\_zhnum\_ganzhi\_cyclic\_mod:nnnn ..... 13, 468, 478  
 \zhnum\_ganzhi\_nian:f ..... 488  
 \zhnum\_ganzhi\_nian:n ..... 14, 489, 505  
 \zhnum\_ganzhi\_normal:nnn ..... 13, 452, 479, 864  
 \c\_zhnum\_hour\_tl ..... 411, 418  
 \c\_zhnum\_hundred\_tl ..... 232, 701  
 \zhnum\_if\_digit:NTF ..... 7, 143, 163, 298  
 \zhnum\_if\_unicode\_engine:TF ..... 18, 771, 775, 880  
 \zhnum\_if\_unicode\_engine\_p: ..... 18, 770, 774, 777  
 \zhnum\_input\_cfg:n ..... 720, 723  
 \zhnum\_int:c ..... 91  
 \zhnum\_int:n 6, 96, 109, 333, 334, 339, 340, 411, 412, 417, 419  
 \zhnum\_integer:n ..... 6, 9, 39, 110  
 \\_zhnum\_integer\_or\_fraction:www ..... 5, 32, 36  
 \l\_zhnum\_last\_cfg\_tl ..... 711, 719, 754  
 \zhnum\_load\_cfg:n ..... 17, 708, 715  
 \zhnum\_load\_cfg:o ..... 817, 828  
 \\_zhnum\_loop\_end:wnn ..... 145, 151  
 \c\_zhnum\_minus\_tl ..... 103, 159, 310  
 \l\_zhnum\_minus\_tl ..... 509  
 \c\_zhnum\_minute\_tl ..... 412, 420  
 \c\_zhnum\_mon\_tl ..... 351

\l_zhnum_mon_tl .....	652	\zhnum_set_financial_map:nn ....	15, 574, 970, 971,
\c_zhnum_month_tl .....	333, 339		972, 973, 979, 981, 982, 988, 990, 991, 992, 993, 994, 995
\l_zhnum_normal_bool .....	632, 838, 843, 848	\zhnum_set_financial_map:nnn	15, 576, 975, 976, 984, 985
\l_zhnum_null_bool .....	646, 853	\zhnum_set_scale:n .....	13, 439, 443
\l_zhnum_null_tl .....	510, 647	\zhnum_set_tiangang_map:nn .....	15,
\zhnum_number:f .....	16, 24		581, 996, 997, 998, 999, 1000, 1001, 1002, 1003, 1004, 1005
\zhnum_number:n .....	4, 27, 35, 53, 72	\zhnum_set_week_day: .....	15, 594, 650
\_zhnum_number:www .....	4, 28, 29	\zhnum_set_zero: .....	15, 593, 642
\_zhnum_output:nnwnn .....	144, 149	\l_zhnum_simp_bool .....	615, 850, 851
\_zhnum_output_digits:NN .....	299, 312	\zhnum_split_number:fn .....	181
\zhnum_parse_config: .....	15, 589, 713, 784	\zhnum_split_number:nn .....	8, 198, 199, 204
\zhnum_parse_number:f .....	99, 104	\zhnum_split_number:NNfNNNNw .....	216, 217
\zhnum_parse_number:n .....	7, 171, 185	\zhnum_split_number:NNnNNNNw .....	8, 201, 205, 220
\zhnum_parse_number:nn .....	7, 160, 172, 173	\_zhnum_split_number_aux:nnn .....	8, 182, 186
\_zhnum_parse_number:nnn .....	174, 175	\_zhnum_split_number_aux:wnn .....	188, 197
\c_zhnum_parts_tl .....	47, 57	\c_zhnum_sun_tl .....	350
\zhnum_process_number:NNNNNN .....	8, 212, 221	\l_zhnum_sun_tl .....	664
\_zhnum_prop_gset_eq:Nn .....	736, 751	\c_zhnum_ten_tl .....	253, 699
\_zhnum_prop_initial:Nn .....	728, 746	\c_zhnum_thousand_tl .....	225, 703
\_zhnum_read_abs_loop:Nw .....	7, 137, 141, 146	\c_zhnum_thu_tl .....	354
\_zhnum_read_digits:w .....	278, 307	\l_zhnum_thu_tl .....	658
\_zhnum_read_digits_loop:NN .....	291, 296, 304	\_zhnum_time:ww .....	403, 409
\_zhnum_read_integer:www .....	7, 112, 153	\l_zhnum_time_bool .....	343, 855, 856
\_zhnum_read_sign_loop:N .....	114, 119, 122	\c_zhnum_tue_tl .....	352
\_zhnum_read_sign_loop:NN .....	280, 284, 287	\l_zhnum_tue_tl .....	654
\_zhnum_read_zeros_loop:N .....	127, 131, 134	\zhnum_two_digits:n .....	11, 361, 372
\l_zhnum_reset_bool .....	595, 598, 727, 755	\zhnum_update_cfg:n .....	712, 716
\zhnum_reset_config: .....	18, 784, 816, 858	\_zhnum_update_cfg_prop:N .....	721, 728, 736, 739
\_zhnum_result:nn .....	117, 149, 150, 151	\c_zhnum_wed_tl .....	353
\c_zhnum_sat_tl .....	349	\l_zhnum_wed_tl .....	656
\l_zhnum_sat_tl .....	662	\_zhnum_week_day:www .....	11, 328, 345, 346
\l_zhnum_scale_int .....	430, 432, 433, 449	\c_zhnum_weekday_tl ..	653, 655, 657, 659, 661, 663, 665
\zhnum_scale_map:n .....	12, 213, 424, 430, 431	\c_zhnum_year_tl .....	332, 338
\zhnum_scale_map_hook:n .....	427, 434, 441	\zhnum_Zeller:nnn .....	11, 348, 358
\zhnum_scale_map_loop:n .....	12, 429, 434	\_zhnum_Zeller_aux:Nnnn .....	362, 363, 366
\zhnum_set_active: .....	787, 788	\zhnum_Zeller_aux:Nnnn .....	11
\zhnum_set_alias: .....	17, 624, 695	\zhnum_Zeller_Gregorian:nnn .....	11, 362, 377
\zhnum_set_alias:NN .....	694, 697, 699, 701, 703, 707	\zhnum_Zeller_Julian:nnn .....	11, 363, 390
\zhnum_set_catcode: .....	18, 730, 778, 786	\zhnum_zero_mod:nn .....	671, 680, 681
\zhnum_set_cfg_name:Nn .....	18, 710, 779, 806	\_zhnum_zero_mod_aux:nn .....	672, 673
\zhnum_set_digits_map:nn .....		\c_zhnum_zero_tl	65, 71, 106, 156, 211, 224, 227, 235, 697
	15, 567, 891, 893, 895, 898, 900, 901,	\zhnumber .....	1, 4, 4, 13
	902, 903, 904, 905, 906, 907, 908, 909, 910, 911, 912,	\zhnumberwithoptions .....	4, 5, 17, 20
	913, 914, 915, 921, 923, 924, 932, 933, 935, 936, 937,	\zhnumExtendScaleMap .....	2, 12, 436
	938, 939, 947, 948, 950, 956, 958, 959, 960, 966, 968, 969	\zhnumsetup .....	3, 20, 871
\zhnum_set_digits_map:nnn	15, 569, 887, 888, 917, 918,	\zhnumwithoptions .....	78, 81
	926, 927, 928, 929, 941, 942, 943, 944, 952, 953, 962, 963	\zhtiangang .....	2, 13, 481
\zhnum_set_dizhi_map:nn .....	15, 583, 1006, 1007,	\zhtime .....	2, 12, 402
	1008, 1009, 1010, 1011, 1012, 1013, 1014, 1015, 1016, 1017	\zhtoday .....	2, 10, 336
		\zhweekday .....	2, 10, 344