

zhnumber 宏包

李清

sobenlee@gmail.com

2014/09/09 v2.0*

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^AT_EX 3 项目的 `expl3`, `xparse` 和 `l3keys2e` 宏包。

2 使用方法

`encoding`

`encoding = {GBK|Big5|UTF8}`

Updated: 2014-09-09

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: 2014-09-09

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^AT_EX 计数器的值转换为中文数字。例如

二

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

<code>\zhweekday</code>	<code>\zhweekday {⟨yyyy/mm/dd⟩}</code>
New: 2012-5-25	输出日期当天的星期。例如
	星期日
	1 <code>\zhweekday{2012/5/20}</code>

<code>\zhdate</code>	<code>\zhdate {⟨yyyy/mm/dd⟩}</code>
<code>\zhdate *</code>	<code>{⟨yyyy/mm/dd⟩}</code>
New: 2012-5-25	以中文格式输出日期, 其中带 <code>*</code> 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 <code>\zhdate{2012/5/21}\</code>
	2 <code>\zhdate*{2012/5/21}</code>

<code>\zhtoday</code>	与 <code>\today</code> 类似, 以中文输出当天的日期。例如
New: 2012-5-25	2014 年 9 月 9 日
	1 <code>\zhtoday</code>

<code>\zhptime</code>	<code>\zhptime {⟨hh:mm⟩}</code>
New: 2012-5-25	以中文格式输出时间。例如
	23 时 56 分
	1 <code>\zhptime{23:56}</code>

<code>\zhcurrtime</code>	输出当前的时间。例如
New: 2012-5-25	22 时 12 分
	1 <code>\zhcurrtime</code>

<code>\zhnumExtendScaleMap</code>	<code>\zhnumExtendScaleMap [⟨character⟩] {⟨character⟩₁, ⟨character⟩₂, ..., ⟨character⟩_n}</code>
New: 2012-5-25	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> 。⟨character _i ⟩ 设置 $10^{4(i+1)}$ 。若给出可选项 ⟨character⟩, 则当数字大于 $10^{4(n+1)} - 1$ 时, 统一用 ⟨character⟩ 设置输出数字的进位。

<code>\zhnumsetup</code>	<code>\zhnumsetup {⟨key⟩=⟨val⟩, ⟨key⟩=⟨val⟩, ...}</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 ⟨key⟩ 如下介绍。

<code>time</code>	<code>time = {⟨Arabic⟩ ⟨Chinese⟩}</code>
New: 2012-5-25	设置日期和时间的数字格式, ⟨Arabic⟩ 为阿拉伯数字, 而 ⟨Chinese⟩ 为中文数字。默认使用阿拉伯数字。例如
	二〇一四年九月九日二十二时十二分
	1 <code>\zhnumsetup{time=Chinese}</code>
	2 <code>\zhtoday\zhcurrtime</code>

<code>style</code>	<code>style = {⟨Simplified⟩ ⟨Traditional⟩ ⟨Normal⟩ ⟨Financial⟩ ⟨Ancient⟩}</code>
Updated: 2012-5-25	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

	1 <code>\zhnumsetup{style={Traditional,Financial}}</code>
	2 <code>\zhnumber{62012.3}\</code>
	3 <code>\zhnumsetup{style=Ancient}</code>
	4 <code>\zhnumber{21}</code>
	陸萬貳仟零壹拾貳點叁
	廿一

`null` `null = <true|false>`

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○,可以使用这个选项。

`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
dot and parts
year month day hour minute weekday mon tue wed thu fri sat sun
```

其中`-`设置负,`-0`设置○,`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 ,而`Fn`设置数字 n 的大写。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: 2014-09-09

`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 \pdfLaTeX 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`

Updated: 2014-09-09

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

3 zhnumber 宏包代码实现

```
1 <*package>
2 <@@=zhnum>
3 \msg_new:nnn { zhnumber } { 13-too-old }
4 {
5   Support~package~'expl3'~too~old. \\\
6   Please~update~an~up~to~date~version~of~the~bundles\\\
7   'l3kernel'~and~'l3packages'\\\
8   using~your~TeX~package~manager~or~from~CTAN.
9 }
10 \ifpackagelater { expl3 } { 2014/08/25 } { }
11 { \msg_error:nn { zhnumber } { 13-too-old } }
12 \RequirePackage { xparse , l3keys2e }
13 \DeclareExpandableDocumentCommand \zhnumber { +o +m }
14 {
15   \IfNoValueTF {#1}
16   { \zhnum_number:n {#2} }
17   { \zhnumberwithoptions {#1} {#2} }
18 }
```

`\zhnumber` 用于将输入的数字按照中文格式输出。

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

`\zhnumberwithoptions` 带选项的用户函数。

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

(End definition for `\zhnumberwithoptions`. This function is documented on page ??.)

`\zhnum_number:n` 先判断输入的是小数还是分数。

```
\__zhnum_number:www 26 \cs_new:Npn \zhnum_number:n #1
27 { \__zhnum_number:www #1 . \q_nil . \q_stop }
28 \cs_new:Npn \__zhnum_number:www #1 . #2 . #3 \q_stop
29 {
30   \quark_if_nil:nTF {#2}
31   { \__zhnum_integer_or_fraction:www #1 / \q_nil / \q_stop }
32   { \zhnum_decimal:nn {#1} {#2} }
33 }
```

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

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

```
34 \cs_new:Npn \__zhnum_integer_or_fraction:www #1 / #2 / #3 \q_stop
35 {
36   \quark_if_nil:nTF {#2}
37   { \zhnum_integer:n {#1} }
38   { \__zhnum_fraction:www #2 \q_mark #1 ; \q_nil ; \q_stop }
39 }
```

(End definition for `__zhnum_integer_or_fraction:www`.)

`__zhnum_fraction:www` 对分数进行预处理。

```
40 \cs_new:Npn \__zhnum_fraction:www #1 \q_mark #2 ; #3 ; #4 \q_stop
41 {
42   \quark_if_nil:nTF {#3}
43   {
44     \zhnum_blank_to_zero:f {#1}
45     \c__zhnum_parts_tl
46     \zhnum_blank_to_zero:f {#2}
47   }
48   {
49     \tl_if_blank:fF {#2}
50     {
51       \zhnum_number:n {#2}
52       \c__zhnum_and_tl
53     }
54     \zhnum_blank_to_zero:f {#1}
55     \c__zhnum_parts_tl
56     \zhnum_blank_to_zero:f {#3}
57   }
58 }
59 \cs_generate_variant:Nn \tl_if_blank:nF { f }
```

(End definition for `__zhnum_fraction:www`.)

`\zhnum_decimal:nn` 对小数进行预处理。

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

(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 }
74 \cs_generate_variant:Nn \zhnum_blank_to_zero:n { f }

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

`\zhnum` 用于将 \LaTeX 计数器按中文格式输出。

```
\zhnumberwithoptions 75 \DeclareExpandableDocumentCommand \zhnum { o m }
76 {
77   \IfNoValueTF {#1}
78     { \zhnum_counter:n {#2} }
79     { \zhnumwithoptions {#1} {#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 3.)

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

```
88 \cs_new_nopar:Npn \zhnum_counter:n #1
89 { \exp_args:Nc \zhnum_counter:Nn { c@#1 } {#1} }
90 \cs_new:Npn \zhnum_counter:Nn #1#2
91 {
92   \token_if_int_register:NTF #1
93     { \zhnum_int:n {#1} }
94     { \__msg_expandable_error:n { `#2' is not a LaTeX counter. } }
95 }
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 }
```

(End definition for `\zhnum_counter: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。

```
109 \cs_new:Npn \zhnum_integer:n #1
110 {
111   \exp_after:wN \__zhnum_read_integer:www
112   \tex_number:D
113   \exp_after:wN \__zhnum_read_sign_loop:N
114   \tex_romannumeral:D -`0 \use:n
115   #1 \exp_stop_f: \q_recursion_tail \q_recursion_stop
116   \__zhnum_result:nn { \c_zero } { } ;
117 }
118 \cs_new:Npn \__zhnum_read_sign_loop:N #1
119 {
120   \if:w + \if:w - \exp_not:N #1 + \fi: \exp_not:N #1
```

```

121     \exp_after:wN \__zhnum_read_sign_loop:N
122     \tex_romannumeral:D -`0 \exp_after:wN \use:n
123   \else:
124     1 \exp_after:wN ;
125     \tex_romannumeral:D -`0
126     \exp_after:wN \__zhnum_read_zeros_loop:N
127     \exp_after:wN #1
128   \fi:
129 }
130 \cs_new:Npn \__zhnum_read_zeros_loop:N #1
131 {
132   \if:w 0 \exp_not:N #1
133     \exp_after:wN \__zhnum_read_zeros_loop:N
134     \tex_romannumeral:D -`0 \exp_after:wN \use:n
135   \else:
136     \exp_after:wN \__zhnum_read_abs_loop:Nw
137     \exp_after:wN #1
138   \fi:
139 }

```

(End definition for \zhnum_integer:n.)

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

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

我们在这里参考 __tl_act:NNNnn 的实现对它进行了改进。

```

140 \cs_new:Npn \__zhnum_read_abs_loop:Nw #1#2 \q_recursion_stop
141 {
142   \zhnum_if_digit:Ntf #1
143   { \__zhnum_output:nnwnn { + \c_one } #1 }
144   { \quark_if_recursion_tail_stop_do:Nn #1 { \__zhnum_loop_end:wnn } }
145   \exp_after:wN \__zhnum_read_abs_loop:Nw
146   \tex_romannumeral:D -`0 \use:n #2 \q_recursion_stop
147 }
148 \cs_new:Npn \__zhnum_output:nnwnn #1#2#3 \__zhnum_result:nn #4#5
149 { #3 \__zhnum_result:nn { #4#1 } { #5#2 } }
150 \cs_new:Npn \__zhnum_loop_end:wnn #1 \__zhnum_result:nn #2#3
151 { \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。

```

152 \cs_new:Npn \__zhnum_read_integer:www #1 ; #2 ; #3 ;
153 {
154   \int_compare:nNtf {#2} = \c_zero
155   { \c__zhnum_zero_tl }
156   {
157     \int_compare:nNf {#1} = \c_one
158     { \c__zhnum_minus_tl }
159     \zhnum_parse_number:nn {#2} {#3}
160   }
161 }

```

(End definition for __zhnum_read_integer:www.)

\zhnum_if_digit:Ntf 判断 #1 是否为数字位。

```

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

```

(End definition for \zhnum_if_digit:Ntf.)

```

\zhnum_parse_number:n
\zhnum_parse_number:nn 170 \cs_new_nopar:Npn \zhnum_parse_number:n #1
171 { \exp_args:Nf \zhnum_parse_number:nn { \tl_count:n {#1} } {#1} }
172 \cs_new_nopar:Npn \zhnum_parse_number:nn #1#2
173 { \exp_args:Nf \__zhnum_parse_number:nnn { \int_mod:nn {#1} \c_four } {#1} {#2} }
174 \cs_new_nopar:Npn \__zhnum_parse_number:nnn #1#2#3
175 {
176   \int_compare:nNnTF {#2} < \c_two
177   { \zhnum_digit_map:n {#3} }
178   {
179     \zhnum_split_number:fNNfn { \zhnum_insert_zeros:n {#1} #3 }
180     \c_true_bool \c_true_bool
181     {
182       \int_compare:nNnTF {#1} = \c_zero
183       { \int_eval:n { (#2) / \c_four - \c_one } }
184       { \int_div_truncate:nn {#2} \c_four }
185     }
186     { \c_zero }
187   }
188 }
189 \cs_generate_variant:Nn \zhnum_parse_number:n { f }

```

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

\zhnum_insert_zeros:n 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。

```

190 \cs_new_nopar:Npn \zhnum_insert_zeros:n #1
191 {
192   \if_case:w \etex_numexpr:D #1 - \c_one \scan_stop:
193   \or: \exp_after:wN \use_none:n
194   \or: \exp_after:wN \use_none:nn
195   \else: \exp_after:wN \use_none:nnn
196   \fi:
197   0 0 0
198 }

```

(End definition for \zhnum_insert_zeros:n.)

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

```

199 \cs_new_nopar:Npn \zhnum_split_number:nNNnn #1#2#3#4#5
200 {
201   \__zhnum_split_number:fnNNnn { \zhnum_number_item:nn {#1} {#5} }
202   {#1} #2#3 {#4} {#5}
203 }
204 \cs_new_nopar:Npn \__zhnum_split_number:nnNNnn #1#2#3#4#5#6
205 {
206   \int_compare:nNnTF {#1} = \c_zero
207   { \__zhnum_split_number_aux:NNfnnn \c_false_bool \c_true_bool }
208   {
209     \bool_if:NF #3 { \c__zhnum_zero_tl }
210     \zhnum_process_number:NNn #3#4 {#1}
211     \exp_args:Nf \zhnum_scale_map:n { \int_eval:n { #5 - #6 } }
212     \int_compare:nNnTF { \int_mod:nn {#1} \c_ten } = \c_zero
213     { \__zhnum_split_number_aux:NNfnnn \c_false_bool \c_true_bool }
214     { \__zhnum_split_number_aux:NNfnnn \c_true_bool \c_false_bool }
215   }
216   { \int_eval:n { #6 + \c_one } } {#2} {#5} {#6}
217 }
218 \cs_new_nopar:Npn \__zhnum_split_number_aux:NNnnnn #1#2#3#4#5#6
219 {
220   \int_compare:nNnF {#5} = {#6}
221   { \zhnum_split_number:nNNnn {#4} #1#2 {#5} {#3} }
222 }
223 \cs_generate_variant:Nn \zhnum_split_number:nNNnn { fNNf }
224 \cs_generate_variant:Nn \__zhnum_split_number:nnNNnn { f }
225 \cs_generate_variant:Nn \__zhnum_split_number_aux:NNnnnn { NNf }

```

(End definition for \zhnum_split_number:nNNnn.)

\zhnum_number_item:nn 截取整数的其中四位数。

```
226 \cs_new_nopar:Npn \zhnum_number_item:nn #1#2
227 { \__zhnum_number_item:nnnnn {#2} #1 \q_recursion_stop }
228 \cs_new_nopar:Npn \__zhnum_number_item:nnnnn #1#2#3#4#5
229 {
230   \int_compare:nNnTF {#1} = \c_zero
231   { \__zhnum_recursion_stop:nnnnw #2#3#4#5 }
232   { \__zhnum_number_item:fnnnn { \int_eval:n { #1 - \c_one } } }
233 }
234 \cs_generate_variant:Nn \__zhnum_number_item:nnnnn { f }
235 \cs_new_nopar:Npn \__zhnum_recursion_stop:nnnnw #1#2#3#4#5 \q_recursion_stop
236 { #1#2#3#4 }
```

(End definition for \zhnum_number_item:nn.)

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

```
237 \cs_new_nopar:Npn \zhnum_process_number:NNn #1#2#3
238 { \zhnum_process_number:nnnnnn #3#1#2 }
239 \cs_new_nopar:Npn \zhnum_process_number:nnnnnn #1#2#3#4#5#6
240 {
241   \int_compare:nNnTF {#1} = \c_zero
242   { \bool_if:NF #6 { \c__zhnum_zero_tl } }
243   { \zhnum_digit_map:n {#1} \c__zhnum_thousand_tl }
244   \int_compare:nNnTF {#2} = \c_zero
245   { \int_compare:nNnT { #1 * (#3#4) } > \c_zero { \c__zhnum_zero_tl } }
246   {
247     \bool_if:nTF
248     { \l__zhnum_ancient_bool && \int_compare_p:nNn {#2} = \c_two }
249     { \zhnum_digit_map:n { #2 00 } }
250     { \zhnum_digit_map:n {#2} \c__zhnum_hundred_tl }
251   }
252   \int_compare:nNnTF {#3} = \c_zero
253   { \int_compare:nNnT { #2 * #4 } > \c_zero { \c__zhnum_zero_tl } }
254   {
255     \bool_if:nF
256     {
257       \int_compare_p:nNn {#3} = \c_one &&
258       \int_compare_p:nNn {#1#2} = \c_zero && #6 && #5
259     }
260     {
261       \bool_if:nTF
262       {
263         \l__zhnum_ancient_bool &&
264         ( \int_compare_p:nNn {#3} = \c_two ||
265           \int_compare_p:nNn {#3} = \c_three ||
266           \int_compare_p:nNn {#3} = \c_four )
267       }
268       { \zhnum_digit_map:n { #3 0 } \use_none:n }
269       { \zhnum_digit_map:n {#3} }
270     }
271     \c__zhnum_ten_tl
272   }
273   \int_compare:nNnF {#4} = \c_zero { \zhnum_digit_map:n {#4} }
274 }
```

(End definition for \zhnum_process_number:NNn and \zhnum_process_number:nnnnnn.)

\zhdigits 将输入的数字输出为中文数字串输出。

```
\zhdigitsoptions 275 \DeclareExpandableDocumentCommand \zhdigits { s o m }
276 {
277   \IfNoValueTF {#2}
278   { \zhnum_digits:Nn #1 {#3} }
279   {
280     \IfBooleanTF #1
281     { \zhdigitsoptions * }
282     { \zhdigitsoptions }
283     {#2} {#3}
284   }
285 }
```



```

286 \NewDocumentCommand \zhdigitswithoptions { s m m }
287 {
288   \group_begin:
289   \keys_set:nn { zhnum / options } {#2}
290   \zhnum_digits:Nn #1 {#3}
291   \group_end:
292 }

```

(End definition for \zhdigits and \zhdigitswithoptions. These functions are documented on page 3.)

\zhnum_digits_zero:n 快捷方式。

```

\zhnum_digits_null:n
293 \cs_new_nopar:Npn \zhnum_digits_zero:n
294 { \zhnum_digits:Nn \BooleanTrue }
295 \cs_new_nopar:Npn \zhnum_digits_null:n
296 { \zhnum_digits:Nn \BooleanFalse }
297 \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。

```

298 \cs_new:Npn \zhnum_digits:Nn #1#2
299 {
300   \exp_after:wN \__zhnum_read_digits:w
301   \tex_number:D
302   \exp_after:wN \__zhnum_read_sign_loop:NN \exp_after:wN #1
303   \tex_romannumeral:D -`0 \use:n
304   #2 \exp_stop_f: \q_recursion_tail \q_recursion_stop
305 }
306 \cs_new:Npn \__zhnum_read_sign_loop:NN #1#2
307 {
308   \if:w + \if:w - \exp_not:N #2 + \fi: \exp_not:N #2
309   \exp_after:wN \__zhnum_read_sign_loop:NN \exp_after:wN #1
310   \tex_romannumeral:D -`0 \exp_after:wN \use:n
311   \else:
312     1 \exp_after:wN ;
313     \exp_after:wN \__zhnum_read_digits_loop:NN
314     \exp_after:wN #1
315     \exp_after:wN #2
316   \fi:
317 }
318 \cs_new:Npn \__zhnum_read_digits_loop:NN #1#2
319 {
320   \zhnum_if_digit:NTF #2
321   { \__zhnum_output_digits:NN #1#2 }
322   {
323     \quark_if_recursion_tail_stop:N #2
324     \if:w .\exp_not:N #2 \exp_after:wN \c__zhnum_dot_tl \fi:
325   }
326   \exp_after:wN \__zhnum_read_digits_loop:NN \exp_after:wN #1
327   \tex_romannumeral:D -`0 \use:n
328 }
329 \cs_new:Npn \__zhnum_read_digits:w #1 ;
330 {
331   \int_compare:nNnF {#1} = \c_one
332   { \c__zhnum_minus_tl }
333 }
334 \cs_new:Npn \__zhnum_output_digits:NN #1#2
335 {
336   \cs:w
337   c__zhnum_
338   \if_int_compare:w #2 = \c_zero
339   \IfBooleanTF #1 { zero } { null }
340   \else:
341     #2
342   \fi:
343   _tl
344   \cs_end:
345 }

```

(End definition for \zhnum_digits:Nn.)

\zhdate 输出中文日期。

```
346 \DeclareExpandableDocumentCommand \zhdate { s m }
347 {
348   \__zhnum_date:www #2 \q_stop
349   \IfBooleanT {#1}
350     { \__zhnum_week_day:www #2 \q_stop }
351 }
352 \cs_new_nopar:Npn \__zhnum_date:www #1/#2/#3 \q_stop
353 {
354   \zhnum_check_time:Nn \zhnum_digits_null:n {#1} \c__zhnum_year_tl
355   \zhnum_check_time:Nn \zhnum_integer:n {#2} \c__zhnum_month_tl
356   \zhnum_check_time:Nn \zhnum_integer:n {#3} \c__zhnum_day_tl
357 }
```

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

\zhtoday 输出当天日期。

```
358 \cs_new_nopar:Npn \zhtoday
359 {
360   \zhnum_check_time:Nn \zhnum_digits_null:V \tex_year:D \c__zhnum_year_tl
361   \zhnum_check_time:Nn \zhnum_int:n \tex_month:D \c__zhnum_month_tl
362   \zhnum_check_time:Nn \zhnum_int:n \tex_day:D \c__zhnum_day_tl
363 }
```

(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。

```
364 \cs_new_nopar:Npn \zhnum_check_time:Nn #1
365 { \bool_if:NTF \l__zhnum_time_bool {#1} { \int_eval:n } }
```

(End definition for \zhnum_check_time:Nn.)

\zhweekday 输出星期

```
366 \cs_new_nopar:Npn \zhweekday #1
367 { \__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 $d = h + 5 \pmod{7} + 1$ 。

```
368 \cs_new_nopar:Npn \__zhnum_week_day:www #1/#2/#3 \q_stop
369 {
370   \if_case:w \etex_numexpr:D \zhnum_Zeller:nnn {#1} {#2} {#3} \scan_stop:
371     \c__zhnum_sat_tl
372   \or: \c__zhnum_sun_tl
373   \or: \c__zhnum_mon_tl
374   \or: \c__zhnum_tue_tl
375   \or: \c__zhnum_wed_tl
376   \or: \c__zhnum_thu_tl
377   \or: \c__zhnum_fri_tl
378   \fi:
379 }
```

(End definition for __zhnum_week_day:www.)

\zhnum_Zeller:nnn 用 Zeller 公式¹ 计算星期几。

```
\zhnum_Zeller_aux:Nnnn
\zhnum_two_digits:n
380 \cs_new_nopar:Npn \zhnum_Zeller:nnn #1#2#3
381 {
382   \int_compare:nNnTF
383     { #1 \zhnum_two_digits:n {#2} \zhnum_two_digits:n {#3} } > { 1582 10 04 }
384     { \__zhnum_Zeller_aux:Nnnn \zhnum_Zeller_Gregorian:nnn }
385     { \__zhnum_Zeller_aux:Nnnn \zhnum_Zeller_Julian:nnn }
386     {#1} {#2} {#3}
387 }
388 \cs_new_nopar:Npn \__zhnum_Zeller_aux:Nnnn #1#2#3#4
389 {
```

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

```

390 \int_compare:nNnTF {#3} < \c_three
391 { #1 { #2 - \c_one } { #3 + \c_twelve } {#4} }
392 { #1 {#2} {#3} {#4} }
393 }
394 \cs_new_nopar:Npn \zhnum_two_digits:n #1
395 {
396 \int_compare:nNnT {#1} < \c_ten { 0 }
397 \int_eval:n {#1}
398 }

```

(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 $Y --$ 。

```

399 \cs_new_nopar:Npn \zhnum_Zeller_Gregorian:nnn #1#2#3
400 {
401 \int_mod:nn
402 {
403 (#3)
404 + \int_div_truncate:nn { 26 * ( #2 + \c_one ) } \c_ten
405 + (#1)
406 + \int_div_truncate:nn {#1} \c_four
407 + \c_six * \int_div_truncate:nn {#1} \c_one_hundred
408 + \int_div_truncate:nn {#1} { 400 }
409 }
410 { \c_seven }
411 }

```

(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

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

```

412 \cs_new_nopar:Npn \zhnum_Zeller_Julian:nnn #1#2#3
413 {
414 \int_mod:nn
415 {
416 (#3)
417 + \int_div_truncate:nn { 26 * ( #2 + \c_one ) } \c_ten
418 + (#1)
419 + \int_div_truncate:nn {#1} \c_four
420 + \c_five
421 }
422 { \c_seven }
423 }

```

(End definition for \zhnum_Zeller_Julian:nnn.)

\zhptime 输出时间。

```

424 \cs_new_nopar:Npn \zhptime #1
425 { \_zhnum_time:ww #1 \q_stop }
426 \group_begin:
427 \char_set_lccode:nn { ` \ ; } { ` \ : }
428 \tl_to_lowercase:n
429 {
430 \group_end:
431 \cs_new_nopar:Npn \_zhnum_time:ww #1 ; #2 \q_stop
432 {
433 \zhnum_check_time:Nn \zhnum_integer:n {#1} \c__zhnum_hour_tl
434 \zhnum_check_time:Nn \zhnum_integer:n {#2} \c__zhnum_minute_tl
435 }
436 }

```

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

\zhcurrtime 输出当前时间。

```
437 \cs_new_nopar:Npn \zhcurrtime
438 {
439   \zhnum_check_time:Nn \zhnum_int:n
440   { \int_div_truncate:nn \tex_time:D { 60 } } \c__zhnum_hour_tl
441   \zhnum_check_time:Nn \zhnum_int:n
442   { \int_mod:nn \tex_time:D { 60 } } \c__zhnum_minute_tl
443 }
```

(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。

```
444 \cs_new_nopar:Npn \zhnum_digit_map:n #1
445 { \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 446 \cs_new_nopar:Npn \zhnum_scale_map:n #1
447 {
448   \cs_if_exist_use:cF { c__zhnum_s #1 _tl }
449   { \zhnum_scale_map_hook:n {#1} }
450 }
451 \cs_new_nopar:Npn \zhnum_scale_map_loop:n #1
452 { \zhnum_scale_map:n { \int_mod:nn {#1} \g__zhnum_scale_int } }
453 \int_new:N \g__zhnum_scale_int
454 \int_set_eq:NN \g__zhnum_scale_int \c_eleven
455 \cs_new_eq:NN \zhnum_scale_map_hook:n \zhnum_scale_map_loop:n
```

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

\zhnumExtendScaleMap

```
456 \NewDocumentCommand \zhnumExtendScaleMap { > { \TrimSpaces } o m }
457 {
458   \int_zero:N \l_tmpa_int
459   \clist_map_inline:nn {#2}
460   {
461     \int_incr:N \l_tmpa_int
462     \tl_set:Nx \l_tmpa_tl
463     { c__zhnum_s \int_eval:n { \l_tmpa_int + \c_eleven } _tl }
464     \tl_if_exist:cF { \l_tmpa_tl }
465     { \int_incr:N \g__zhnum_scale_int }
466     \tl_set:cn { \l_tmpa_tl } {##1}
467   }
468   \IfNoValueF {#1}
469   { \cs_set:Npn \zhnum_scale_map_hook:n ##1 {#1} }
470 }
```

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

根据需要设置中文阿拉伯数字。

```
471 \group_begin:
472   \tl_set:Nn \l_tmpa_tl
473   {
474     - .tl_set:N = \l__zhnum_minus_tl ,
475     -0 .tl_set:N = \l__zhnum_null_tl ,
476   }
477   \tl_put_right:Nx \l_tmpa_tl
478   {
479     E2 .tl_set:N = \exp_not:c { l__zhnum_ 100 _tl } ,
480     E3 .tl_set:N = \exp_not:c { l__zhnum_ 1000 _tl } ,
481     FE2 .tl_set:N = \exp_not:c { l__zhnum_financial_ 100 _tl } ,
482     FE3 .tl_set:N = \exp_not:c { l__zhnum_financial_ 1000 _tl } ,
483   }
484   \clist_map_inline:nn
485   { 0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 100 , 1000 }
486   {
```

```

487     \tl_put_right:Nx \l_tmpa_tl
488     {
489         #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } ,
490         F#1 .tl_set:N = \exp_not:c { l__zhnum_financial_ #1 _tl } ,
491     }
492 }
493 \clist_map_inline:nn
494 { 20 , 30 , 40 , 200 }
495 {
496     \tl_put_right:Nx \l_tmpa_tl
497     { #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } , }
498 }
499 \clist_map_inline:nn
500 { 4 , 8 , 12 , 16 , 20 , 24 , 28 , 32 , 36 , 40 , 44 }
501 {
502     \tl_put_right:Nx \l_tmpa_tl
503     { E#1 .tl_set:N = \exp_not:c { l__zhnum_s \int_eval:n { #1 / 4 } _tl } , }
504 }
505 \clist_map_inline:nn
506 {
507     dot , and , parts , year , month , day , weekday , hour , minute
508     mon , tue , wed , thu , fri , sat , sun
509 }
510 {
511     \tl_put_right:Nx \l_tmpa_tl
512     { #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } , }
513 }
514 \use:x
515 {
516     \group_end:
517     \keys_define:nn { zhnum / options } { \exp_not:o \l_tmpa_tl }
518 }

```

将配置文件中的中文数字保存到 prop 变量中。

```

\zhnum_set_digits_map:nn 将配置文件中的中文数字保存到 prop 变量中。
\zhnum_set_digits_map:nnn 519 \cs_new_protected_nopar:Npn \zhnum_set_digits_map:nn #1#2
\zhnum_set_financial_map:nn 520 { \prop_put:Nnn \l__zhnum_cfg_map_prop {#1} {#2} }
\zhnum_set_financial_map:nnn 521 \cs_new_protected_nopar:Npn \zhnum_set_digits_map:nnn #1#2#3
522 {
523     \prop_put_if_new:Nnn \l__zhnum_cfg_map_prop {#1} {#3}
524     \prop_put:Nnn \l__zhnum_cfg_map_var_prop {#1_#2} {#3}
525 }
526 \cs_new_protected_nopar:Npn \zhnum_set_financial_map:nn #1#2
527 { \prop_put:Nnn \l__zhnum_cfg_map_finan_prop {#1} {#2} }
528 \cs_new_protected_nopar:Npn \zhnum_set_financial_map:nnn #1#2#3
529 {
530     \prop_put_if_new:Nnn \l__zhnum_cfg_map_finan_prop {#1} {#3}
531     \prop_put:Nnn \l__zhnum_cfg_map_var_prop { financial_#1_#2 } {#3}
532 }
533 \prop_new:N \l__zhnum_cfg_map_prop
534 \prop_new:N \l__zhnum_cfg_map_var_prop
535 \prop_new:N \l__zhnum_cfg_map_finan_prop

```

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

将 prop 表转化到单独的 tl 变量。

```

\zhnum_parse_config: 将 prop 表转化到单独的 tl 变量。
\zhnum_check_simp:nn 536 \cs_new_protected_nopar:Npn \zhnum_parse_config:
\zhnum_check_financial:nn 537 {
538     \prop_map_function:NN \l__zhnum_cfg_map_prop \zhnum_check_simp:nn
539     \bool_if:NF \l__zhnum_reset_bool
540     { \prop_map_function:NN \l__zhnum_cfg_map_prop \zhnum_check_financial:nn }
541     \zhnum_set_zero:
542     \zhnum_set_week_day:
543 }
544 \cs_new_protected_nopar:Npn \zhnum_check_simp:nn #1#2
545 {
546     \__zhnum_check_simp_aux:nn {#1} {#2}
547     \prop_get:NnNT \l__zhnum_cfg_map_finan_prop {#1} \l_tmpa_tl
548     { \exp_args:Nno \__zhnum_check_simp_aux:nn { financial_ #1 } \l_tmpa_tl }
549 }

```

```

550 \cs_new_protected_nopar:Npn \__zhnum_check_simp_aux:nn #1#2
551 {
552   \prop_get:NnNTF \l__zhnum_cfg_map_var_prop { #1 _trad } \l_tmpa_tl
553   {
554     \prop_get:NnNTF \l__zhnum_cfg_map_var_prop { #1 _simp } \l_tmpb_tl
555     {
556       \tl_set:cx { l__zhnum_ #1 _tl }
557       {
558         \exp_not:n { \bool_if:NTF \l__zhnum_simp_bool }
559         { \exp_not:o \l_tmpb_tl } { \exp_not:o \l_tmpa_tl }
560       }
561     }
562     {
563       \tl_set:cx { l__zhnum_ #1 _tl }
564       {
565         \exp_not:n { \bool_if:NTF \l__zhnum_simp_bool }
566         { \exp_not:n {#2} } { \exp_not:o \l_tmpa_tl }
567       }
568     }
569   }
570   { \tl_set:cn { l__zhnum_ #1 _tl } {#2} }
571 }
572 \cs_new_protected_nopar:Npn \zhnum_check_financial:nn #1#2
573 {
574   \prop_get:NnNTF \l__zhnum_cfg_map_finan_prop {#1} \l_tmpa_tl
575   {
576     \zhnum_assgin_const_tl:cx { c__zhnum_ #1 _tl }
577     {
578       \exp_not:n { \bool_if:NTF \l__zhnum_normal_bool }
579       { \exp_not:c { l__zhnum_ #1 _tl } }
580       { \exp_not:c { l__zhnum_financial_ #1 _tl } }
581     }
582   }
583   {
584     \zhnum_assgin_const_tl:cx
585     { c__zhnum_ #1 _tl } { \exp_not:c { l__zhnum_ #1 _tl } }
586   }
587 }
588 \cs_new_protected_nopar:Npn \zhnum_set_zero:
589 {
590   \tl_set:cx { l__zhnum_0_tl }
591   {
592     \exp_not:n { \bool_if:NTF \l__zhnum_null_bool }
593     { \exp_not:o \l__zhnum_null_tl } { \exp_not:v { l__zhnum_0_tl } }
594   }
595 }
596 \cs_new_protected_nopar:Npn \zhnum_set_week_day:
597 {
598   \tl_set:Nx \l__zhnum_mon_tl
599   { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_1_tl } }
600   \tl_set:Nx \l__zhnum_tue_tl
601   { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_2_tl } }
602   \tl_set:Nx \l__zhnum_wed_tl
603   { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_3_tl } }
604   \tl_set:Nx \l__zhnum_thu_tl
605   { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_4_tl } }
606   \tl_set:Nx \l__zhnum_fri_tl
607   { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_5_tl } }
608   \tl_set:Nx \l__zhnum_sat_tl
609   { \exp_not:N \c__zhnum_weekday_tl \exp_not:v { l__zhnum_6_tl } }
610   \tl_set:Nx \l__zhnum_sun_tl
611   { \exp_not:N \c__zhnum_weekday_tl \exp_not:o \l__zhnum_day_tl }
612   \bool_if:NF \l__zhnum_reset_bool
613   {
614     \clist_map_inline:nn { mon , tue , wed , thu , fri , sat , sun }
615     {
616       \zhnum_assgin_const_tl:cx
617       { c__zhnum_ ##1 _tl } { \exp_not:c { l__zhnum_ ##1 _tl } }
618     }
619   }

```

```

619         \bool_set_true:N \l__zhnum_reset_bool
620     }
621 }
622 \cs_new_eq:NN \zhnum_assgin_const_tl:cx \tl_const:cx
623 \AtEndOfPackage
624 { \cs_set_eq:NN \zhnum_assgin_const_tl:cx \tl_set:cx }

```

(End definition for \zhnum_parse_config: and others.)

\zhnum_set_alias: 一些易于使用的别名。

```

625 \cs_new_protected_nopar:Npn \zhnum_set_alias:
626 {
627     \tl_const:Nx \c__zhnum_zero_tl { \exp_not:c { c__zhnum_ 0 _tl } }
628     \tl_const:Nx \c__zhnum_ten_tl { \exp_not:c { c__zhnum_ 10 _tl } }
629     \tl_const:Nx \c__zhnum_hundred_tl { \exp_not:c { c__zhnum_ 100 _tl } }
630     \tl_const:Nx \c__zhnum_thousand_tl { \exp_not:c { c__zhnum_ 1000 _tl } }
631 }
632 \AtEndOfPackage { \zhnum_set_alias: }

```

(End definition for \zhnum_set_alias:.)

\zhnum_load_cfg:n 根据选定编码载入配置文件。

```

633 \cs_new_protected_nopar:Npn \zhnum_load_cfg:n #1
634 {
635     \zhnum_set_cfg_name:Nn \l__zhnum_cfg_tl {#1}
636     \tl_if_eq:NNF \l__zhnum_cfg_tl \l__zhnum_last_cfg_tl
637     { \zhnum_update_cfg:n {#1} }
638     \zhnum_parse_config:
639 }
640 \cs_generate_variant:Nn \zhnum_load_cfg:n { o }
641 \cs_new_protected_nopar:Npn \zhnum_update_cfg:n #1
642 {
643     \prop_if_exist:CTF { g__zhnum_cfg_ \l__zhnum_cfg_tl _prop }
644     { \tl_set_eq:NN \l__zhnum_last_cfg_tl \l__zhnum_cfg_tl }
645     { \zhnum_input_cfg:n {#1} }
646     \__zhnum_update_cfg_prop:N \prop_set_eq:Nc
647 }
648 \cs_new_protected_nopar:Npn \zhnum_input_cfg:n #1
649 {
650     \file_if_exist_input:nTF { zhnumber - #1 .cfg }
651     {
652         \bool_set_false:N \l__zhnum_reset_bool
653         \__zhnum_update_cfg_prop:N \__zhnum_prop_initial:Nn
654         \group_begin:
655             \zhnum_set_catcode:
656         }
657     {
658         \msg_error:nnx { zhnumber } { file-not-found } {#1}
659         \use_none:nnn
660     }
661     \__zhnum_update_cfg_prop:N \__zhnum_prop_gset_eq:Nn
662     \group_end:
663 }
664 \cs_new_protected_nopar:Npn \__zhnum_update_cfg_prop:N #1
665 {
666     #1 \l__zhnum_cfg_map_prop { g__zhnum_cfg_ \l__zhnum_cfg_tl _prop }
667     #1 \l__zhnum_cfg_map_var_prop { g__zhnum_cfg_var_ \l__zhnum_cfg_tl _prop }
668     #1 \l__zhnum_cfg_map_finan_prop { g__zhnum_cfg_finan_ \l__zhnum_cfg_tl _prop }
669 }
670 \cs_new_protected:Npn \__zhnum_prop_initial:Nn #1#2
671 {
672     \prop_clear:N #1
673     \prop_new:c {#2}
674 }
675 \cs_new_protected:Npn \__zhnum_prop_gset_eq:Nn #1#2
676 { \prop_gset_eq:cN {#2} #1 }
677 \tl_new:N \l__zhnum_cfg_tl
678 \tl_new:N \l__zhnum_last_cfg_tl
679 \bool_new:N \l__zhnum_reset_bool

```

```

680 \msg_new:nnnn { zhnumber } { file-not-found }
681 { File~`#1'~not~found. }
682 {
683   The~requested~file~could~not~be~found~in~the~current~directory,~
684   in~the~TeX~search~path~or~in~the~LaTeX~search~path.
685 }

```

(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。

```

686 \pdfTeX_if_engine:TF
687 {
688   \str_if_eq_x:nnTF
689   { \token_to_str:N \kchar }
690   { \token_to_meaning:N \kchar }
691 }
692 { \use_i:nn }
693 {
694   \cs_new_eq:NN \zhnum_if_unicode_engine_p: \c_true_bool
695   \cs_new_eq:NN \zhnum_if_unicode_engine:TF \use_i:nn
696 }
697 {
698   \cs_new_eq:NN \zhnum_if_unicode_engine_p: \c_false_bool
699   \cs_new_eq:NN \zhnum_if_unicode_engine:TF \use_ii:nn
700 }

```

(End definition for \zhnum_if_unicode_engine:TF.)

\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 701 \if_predicate:w \zhnum_if_unicode_engine_p:
\zhnum_reset_config: 702 \cs_new_eq:NN \zhnum_set_catcode: \prg_do_nothing:
703 \cs_new_protected_nopar:Npn \zhnum_set_cfg_name:Nn #1#2
704 {
705   \tl_set:Nx \l__zhnum_encoding_tl {#2}
706   \tl_set:Nx #1 { \tl_to_str:N \l__zhnum_encoding_tl }
707 }
708 \cs_new_eq:NN \zhnum_reset_config: \zhnum_parse_config:
709 \else:
710 \cs_new_protected_nopar:Npn \zhnum_set_catcode:
711 { \bool_if:NT \l__zhnum_active_char_bool { \zhnum_set_active: } }
712 \cs_new_protected_nopar:Npn \zhnum_set_active:
713 {
714   \str_case:onTF { \l__zhnum_encoding_tl }
715   {
716     { gbk } { \int_set:Nn \l__zhnum_byte_min_int { "81 } }
717     { big5 } { \int_set:Nn \l__zhnum_byte_min_int { "A1 } }
718   }
719   { \int_set:Nn \l__zhnum_byte_max_int { "FE } }
720   {
721     \int_set:Nn \l__zhnum_byte_min_int { "E0 }
722     \int_set:Nn \l__zhnum_byte_max_int { "EF }
723   }
724   \int_step_function:nnnN
725   { \l__zhnum_byte_min_int } { \c_one }
726   { \l__zhnum_byte_max_int } \char_set_catcode_active:n
727 }
728 \int_new:N \l__zhnum_byte_min_int
729 \int_new:N \l__zhnum_byte_max_int
730 \cs_new_protected_nopar:Npn \zhnum_set_cfg_name:Nn #1#2
731 {
732   \tl_set:Nx \l__zhnum_encoding_tl {#2}
733   \tl_set:Nx #1
734   {
735     \tl_to_str:N \l__zhnum_encoding_tl
736     \bool_if:NT \l__zhnum_active_char_bool
737     { \tl_to_str:n { _active } }
738   }
739 }

```



```

740 \cs_new_protected_nopar:Npn \zhnum_reset_config:
741 { \zhnum_load_cfg:o { \l__zhnum_encoding_tl } }
742 \bool_new:N \l__zhnum_active_char_bool
743 \bool_set_true:N \l__zhnum_active_char_bool
744 \fi:

```

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

encoding 宏包设置选项。

```

style 745 \keys_define:nn { zhnum / options }
null 746 {
reset 747   encoding .choices:nn =
748     { UTF8 , GBK , Big5 }
749     {
750       \tl_set:Nx \l__zhnum_encoding_tl
751       { \exp_args:No \tl_expandable_lowercase:n { \l_keys_choice_tl } }
752       \zhnum_load_cfg:o { \l__zhnum_encoding_tl }
753     } ,
754   encoding .default:n = { GBK } ,
755   encoding / Bg5 .meta:n = { encoding = Big5 } ,
756   encoding / unknown .code:n =
757     { \msg_error:nnn { zhnumber } { encoding-invalid } {#1} } ,
758   style .multichoice: ,
759   style / Normal .code:n =
760     {
761       \bool_set_false:N \l__zhnum_ancient_bool
762       \bool_set_true:N \l__zhnum_normal_bool
763     } ,
764   style / Financial .code:n =
765     {
766       \bool_set_false:N \l__zhnum_ancient_bool
767       \bool_set_false:N \l__zhnum_normal_bool
768     } ,
769   style / Ancient .code:n =
770     {
771       \bool_set_true:N \l__zhnum_ancient_bool
772       \bool_set_true:N \l__zhnum_normal_bool
773     } ,
774   style / Simplified .code:n = { \bool_set_true:N \l__zhnum_simp_bool } ,
775   style / Traditional .code:n = { \bool_set_false:N \l__zhnum_simp_bool } ,
776   style .default:n = { Normal , Simplified } ,
777   null .bool_set:N = \l__zhnum_null_bool ,
778   time .choice: ,
779   time / Chinese .code:n = { \bool_set_true:N \l__zhnum_time_bool } ,
780   time / Arabic .code:n = { \bool_set_false:N \l__zhnum_time_bool } ,
781   time .default:n = { Arabic } ,
782   reset .code:n = { \zhnum_reset_config: } ,
783   activechar .bool_set:N = \l__zhnum_active_char_bool ,
784 }
785 \tl_new:N \l__zhnum_encoding_tl
786 \msg_new:nnnn { zhnumber } { encoding-invalid }
787 { The~encoding~`#1'~is~invalid. }
788 { Available~encoding~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。

```

789 \NewDocumentCommand \zhnumsetup { m }
790 {
791   \keys_set:nn { zhnum / options } {#1}
792   \tex_ignorespaces:D
793 }

```

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

初始化设置和执行宏包选项。

```

794 \keys_set:nn { zhnum / options } { style , time }
795 \ProcessKeysOptions { zhnum / options }

```

如果没有选定编码,则根据引擎自动设置编码。

```
796 \tl_if_empty:NT \l__zhnum_encoding_tl
797 {
798   \zhnum_if_unicode_engine:TF
799     { \keys_set:nn { zhnum / options } { encoding = UTF8 } }
800     { \keys_set:nn { zhnum / options } { encoding = GBK } }
801 }
802 </package>
```

4 中文数字配置文件

```
803 <*config>
804 <!*big5>
805 \zhnum_set_digits_map:nnn { minus } { simp } { 负 }
806 \zhnum_set_digits_map:nnn { minus } { trad } { 負 }
807 </!big5>
808 <*big5>
809 \zhnum_set_digits_map:nn { minus } { 負 }
810 </big5>
811 \zhnum_set_digits_map:nn { 0 } { 零 }
812 <!*big5>
813 \zhnum_set_digits_map:nn { null } { 〇 }
814 </!big5>
815 <*big5>
816 \zhnum_set_digits_map:nn { null } { 〇 }
817 </big5>
818 \zhnum_set_digits_map:nn { 1 } { 一 }
819 \zhnum_set_digits_map:nn { 2 } { 二 }
820 \zhnum_set_digits_map:nn { 3 } { 三 }
821 \zhnum_set_digits_map:nn { 4 } { 四 }
822 \zhnum_set_digits_map:nn { 5 } { 五 }
823 \zhnum_set_digits_map:nn { 6 } { 六 }
824 \zhnum_set_digits_map:nn { 7 } { 七 }
825 \zhnum_set_digits_map:nn { 8 } { 八 }
826 \zhnum_set_digits_map:nn { 9 } { 九 }
827 \zhnum_set_digits_map:nn { 10 } { 十 }
828 \zhnum_set_digits_map:nn { 100 } { 百 }
829 \zhnum_set_digits_map:nn { 1000 } { 千 }
830 \zhnum_set_digits_map:nn { 20 } { 廿 }
831 \zhnum_set_digits_map:nn { 30 } { 卅 }
832 \zhnum_set_digits_map:nn { 40 } { 卌 }
833 \zhnum_set_digits_map:nn { 200 } { 貳佰 }
834 <!*big5>
835 \zhnum_set_digits_map:nnn { dot } { simp } { 点 }
836 \zhnum_set_digits_map:nnn { dot } { trad } { 點 }
837 </!big5>
838 <*big5>
839 \zhnum_set_digits_map:nn { dot } { 點 }
840 </big5>
841 \zhnum_set_digits_map:nn { and } { 又 }
842 \zhnum_set_digits_map:nn { parts } { 分之 }
843 \zhnum_set_digits_map:nn { s0 } { }
844 <!*big5>
845 \zhnum_set_digits_map:nnn { s1 } { simp } { 万 }
846 \zhnum_set_digits_map:nnn { s1 } { trad } { 萬 }
847 \zhnum_set_digits_map:nnn { s2 } { simp } { 亿 }
848 \zhnum_set_digits_map:nnn { s2 } { trad } { 億 }
849 </!big5>
850 <*big5>
851 \zhnum_set_digits_map:nn { s1 } { 萬 }
852 \zhnum_set_digits_map:nn { s2 } { 億 }
853 </big5>
854 \zhnum_set_digits_map:nn { s3 } { 兆 }
855 \zhnum_set_digits_map:nn { s4 } { 京 }
856 \zhnum_set_digits_map:nn { s5 } { 垓 }
857 \zhnum_set_digits_map:nn { s6 } { 秭 }
858 \zhnum_set_digits_map:nn { s7 } { 穰 }
```

```

859 (*!big5)
860 \zhnum_set_digits_map:nnn { s8 } { simp } { 沟 }
861 \zhnum_set_digits_map:nnn { s8 } { trad } { 溝 }
862 \zhnum_set_digits_map:nnn { s9 } { simp } { 洄 }
863 \zhnum_set_digits_map:nnn { s9 } { trad } { 澗 }
864 (/!big5)
865 (*big5)
866 \zhnum_set_digits_map:nn { s8 } { 溝 }
867 \zhnum_set_digits_map:nn { s9 } { 澗 }
868 (/big5)
869 \zhnum_set_digits_map:nn { s10 } { 正 }
870 (*!big5)
871 \zhnum_set_digits_map:nnn { s11 } { simp } { 載 }
872 \zhnum_set_digits_map:nnn { s11 } { trad } { 載 }
873 (/!big5)
874 (*big5)
875 \zhnum_set_digits_map:nn { s11 } { 載 }
876 (/big5)
877 \zhnum_set_digits_map:nn { year } { 年 }
878 \zhnum_set_digits_map:nn { month } { 月 }
879 \zhnum_set_digits_map:nn { day } { 日 }
880 (*!big5)
881 \zhnum_set_digits_map:nnn { hour } { simp } { 時 }
882 \zhnum_set_digits_map:nnn { hour } { trad } { 時 }
883 (/!big5)
884 (*big5)
885 \zhnum_set_digits_map:nn { hour } { 時 }
886 (/big5)
887 \zhnum_set_digits_map:nn { minute } { 分 }
888 \zhnum_set_digits_map:nn { weekday } { 星期 }
889 \zhnum_set_financial_map:nn { null } { 零 }
890 \zhnum_set_financial_map:nn { 0 } { 零 }
891 \zhnum_set_financial_map:nn { 1 } { 壹 }
892 \zhnum_set_financial_map:nn { 2 } { 貳 }
893 (*!big5)
894 \zhnum_set_financial_map:nnn { 3 } { simp } { 叁 }
895 \zhnum_set_financial_map:nnn { 3 } { trad } { 叁 }
896 (/!big5)
897 (*big5)
898 \zhnum_set_financial_map:nn { 3 } { 參 }
899 (/big5)
900 \zhnum_set_financial_map:nn { 4 } { 肆 }
901 \zhnum_set_financial_map:nn { 5 } { 伍 }
902 (*!big5)
903 \zhnum_set_financial_map:nnn { 6 } { simp } { 陆 }
904 \zhnum_set_financial_map:nnn { 6 } { trad } { 陸 }
905 (/!big5)
906 (*big5)
907 \zhnum_set_financial_map:nn { 6 } { 陸 }
908 (/big5)
909 \zhnum_set_financial_map:nn { 7 } { 柒 }
910 \zhnum_set_financial_map:nn { 8 } { 捌 }
911 \zhnum_set_financial_map:nn { 9 } { 玖 }
912 \zhnum_set_financial_map:nn { 10 } { 拾 }
913 \zhnum_set_financial_map:nn { 100 } { 佰 }
914 \zhnum_set_financial_map:nn { 1000 } { 仟 }
915 (/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		etex commands:	
<code>\:</code>	427	<code>\etex_numexpr:D</code>	192, 370
<code>\;</code>	427	exp commands:	
<code>\\</code>	5, 6, 7	<code>\exp_after:wN</code>	111, 113, 121, 122, 124, 126, 127, 133, 134, 136, 137, 145, 165, 167, 193, 194, 195, 300, 302, 309, 310, 312, 313, 314, 315, 324, 326
A		<code>\exp_args:Nc</code>	89
<code>activechar</code>	3	<code>\exp_args:Nf</code>	171, 173, 211
<code>\AtEndOfPackage</code>	623, 632	<code>\exp_args:Nno</code>	548
B		<code>\exp_args:No</code>	751
bingint commands:		<code>\exp_not:c</code>	479, 480, 481, 482, 489, 490, 497, 503, 512, 579, 580, 585, 617, 627, 628, 629, 630
<code>__bingint_read_do:nn</code>	5	<code>\exp_not:N</code>	120, 132, 164, 308, 324, 599, 601, 603, 605, 607, 609, 611
bool commands:		<code>\exp_not:n</code>	558, 565, 566, 578, 592
<code>\bool_if:Nf</code>	209, 242, 539, 612	<code>\exp_not:o</code>	517, 559, 566, 593, 611
<code>\bool_if:nf</code>	255	<code>\exp_not:v</code>	593, 599, 601, 603, 605, 607, 609
<code>\bool_if:NT</code>	711, 736	<code>\exp_stop_f:</code>	115, 164, 304
<code>\bool_if:NTF</code>	365, 558, 565, 578	F	
<code>\bool_if:nTF</code>	247, 261, 592	false commands:	
<code>\bool_new:N</code>	679, 742	<code>\c_false_bool</code>	207, 213, 214, 698
<code>\bool_set_false:N</code>	652, 761, 766, 767, 775, 780	fi commands:	
<code>\bool_set_true:N</code>	619, 743, 762, 771, 772, 774, 779	<code>\fi:</code>	120, 128, 138, 168, 196, 308, 316, 324, 342, 378, 744
<code>\BooleanFalse</code>	296	file commands:	
<code>\BooleanTrue</code>	294	<code>\file_if_exist_input:nTF</code>	650
C		five commands:	
char commands:		<code>\c_five</code>	420
<code>\char_set_catcode_active:n</code>	726	four commands:	
<code>\char_set_lccode:nn</code>	427	<code>\c_four</code>	173, 183, 184, 266, 406, 419
clist commands:		G	
<code>\clist_map_inline:nn</code>	459, 484, 493, 499, 505, 614	group commands:	
cs commands:		<code>\group_begin:</code>	21, 83, 288, 426, 471, 654
<code>\cs:w</code>	336	<code>\group_end:</code>	24, 86, 291, 430, 516, 662
<code>\cs_end:</code>	344	I	
<code>\cs_generate_variant:Nn</code>	59, 67, 74, 189, 223, 224, 225, 234, 297, 640	if commands:	
<code>\cs_if_exist_use:cF</code>	448	<code>\if:w</code>	120, 132, 308, 324
<code>\cs_new:Npn</code>	26, 28, 34, 40, 60, 68, 90, 96, 109, 118, 130, 140, 148, 150, 152, 162, 298, 306, 318, 329, 334	<code>\if_case:w</code>	192, 370
<code>\cs_new_eq:NN</code>	455, 622, 694, 695, 698, 699, 702, 708	<code>\if_int_compare:w</code>	164, 338
<code>\cs_new_nopar:Npn</code>	88, 170, 172, 174, 190, 199, 204, 218, 226, 228, 235, 237, 239, 293, 295, 352, 358, 364, 366, 368, 380, 388, 394, 399, 412, 424, 431, 437, 444, 446, 451	<code>\if_predicate:w</code>	701
<code>\cs_new_protected:Npn</code>	670, 675	<code>\IfBooleanT</code>	349
<code>\cs_new_protected_nopar:Npn</code>	519, 521, 526, 528, 536, 544, 550, 572, 588, 596, 625, 633, 641, 648, 664, 703, 710, 712, 730, 740	<code>\IfBooleanTF</code>	280, 339
<code>\cs_set:Npn</code>	469	<code>\IfNoValueF</code>	468
<code>\cs_set_eq:NN</code>	624	<code>\IfNoValueTF</code>	15, 77, 277
D		int commands:	
<code>\DeclareExpandableDocumentCommand</code>	13, 75, 275, 346	<code>\int_compare:nNfF</code>	157, 220, 273, 331
E		<code>\int_compare:nNtT</code>	245, 253, 396
eleven commands:		<code>\int_compare:nNtTF</code>	98, 101, 154, 176, 182, 206, 212, 230, 241, 244, 252, 382, 390
<code>\c_eleven</code>	454, 463	<code>\int_compare_p:nNn</code>	248, 257, 258, 264, 265, 266
else commands:		<code>\int_div_truncate:nn</code>	184, 404, 406, 407, 408, 417, 419, 440
<code>\else:</code>	123, 135, 166, 195, 311, 340, 709	<code>\int_eval:n</code>	99, 104, 151, 183, 211, 216, 232, 365, 397, 463, 503
encoding		<code>\int_incr:N</code>	461, 465
	1, 17	<code>\int_mod:nn</code>	173, 212, 401, 414, 442, 452
		<code>\int_new:N</code>	453, 728, 729

<code>\int_set:Nn</code>	716, 717, 719, 721, 722		
<code>\int_set_eq:NN</code>	454		
<code>\int_step_function:nnnN</code>	724		
<code>\int_zero:N</code>	458		
		K	
<code>\kchar</code>	689, 690		
keys commands:			
<code>\l_keys_choice_tl</code>	751		
<code>\keys_define:nn</code>	517, 745		
<code>\keys_set:nn</code>	22, 84, 289, 791, 794, 799, 800		
		M	
mark commands:			
<code>\q_mark</code>	38, 40		
msg commands:			
<code>\msg_error:nn</code>	11		
<code>\msg_error:nnn</code>	757		
<code>\msg_error:nnx</code>	658		
<code>_msg_expandable_error:n</code>	94		
<code>\msg_new:nnn</code>	3		
<code>\msg_new:nnnn</code>	680, 786		
		N	
<code>\NewDocumentCommand</code>	19, 81, 286, 456, 789		
nil commands:			
<code>\q_nil</code>	27, 31, 38		
nine commands:			
<code>\c_nine</code>	164		
null	3, 17		
		O	
one commands:			
<code>\c_one</code>	143, 157, 183, 192, 216, 232, 257, 331, 391, 404, 417, 725		
<code>\c_one_hundred</code>	407		
or commands:			
<code>\or:</code>	193, 194, 372, 373, 374, 375, 376, 377		
		P	
pdftex commands:			
<code>\pdfTeX_if_engine:TF</code>	686		
prg commands:			
<code>\prg_do_nothing:</code>	702		
<code>\ProcessKeysOptions</code>	795		
prop commands:			
<code>\prop_clear:N</code>	672		
<code>\prop_get:NnNT</code>	547		
<code>\prop_get:NnNTF</code>	552, 554, 574		
<code>\prop_gset_eq:cN</code>	676		
<code>\prop_if_exist:cTF</code>	643		
<code>\prop_map_function:NN</code>	538, 540		
<code>\prop_new:c</code>	673		
<code>\prop_new:N</code>	533, 534, 535		
<code>\prop_put:Nnn</code>	520, 524, 527, 531		
<code>\prop_put_if_new:Nnn</code>	523, 530		
<code>\prop_set_eq:Nc</code>	646		
		Q	
quark commands:			
<code>\quark_if_nil:nTF</code>	30, 36, 42		
<code>\quark_if_recursion_tail_stop:N</code>	323		
<code>\quark_if_recursion_tail_stop_do:Nn</code>	144		
		R	
recursion commands:			
<code>\q_recursion_stop</code>	115, 140, 146, 227, 235, 304		
<code>\q_recursion_tail</code>	115, 304		
<code>\RequirePackage</code>	12		
reset	3, 17		
		S	
scan commands:			
<code>\scan_stop:</code>	192, 370		
seven commands:			
<code>\c_seven</code>	410, 422		
six commands:			
<code>\c_six</code>	407		
stop commands:			
<code>\q_stop</code> 27, 28, 31, 34, 38, 40, 348, 350, 352, 367, 368, 425, 431			
str commands:			
<code>\str_case:onTF</code>	714		
<code>\str_if_eq_x:nnTF</code>	688		
style	2, 17		
		T	
ten commands:			
<code>\c_ten</code>	212, 396, 404, 417		
TeX and L ^A T _E X 2 _ε commands:			
<code>\@ifpackagelater</code>	10		
<code>\kchar</code>	16		
<code>\zhdate</code>	2, 2		
<code>\zhdigits</code>	1, 1, 1, 1, 2, 3, 3		
<code>\zhnum</code>	1, 1, 3		
<code>\zhnumber</code>	1, 1, 2, 2, 3		
<code>\zhnumExtendScaleMap</code>	2, 2		
<code>\zhnumsetup</code>	1, 2, 3		
<code>\zhtime</code>	2		
<code>\zhweekday</code>	2		
tex commands:			
<code>\tex_day:D</code>	362		
<code>\tex_ignorespaces:D</code>	792		
<code>\tex_month:D</code>	361		
<code>\tex_number:D</code>	112, 301		
<code>\tex_romannumeral:D</code> 114, 122, 125, 134, 146, 303, 310, 327			
<code>\tex_time:D</code>	440, 442		
<code>\tex_year:D</code>	360		
three commands:			
<code>\c_three</code>	265, 390		
time	2		
tl commands:			
<code>_tl_act:NNNnn</code>	6		
<code>\tl_const:cx</code>	622		
<code>\tl_const:Nx</code>	627, 628, 629, 630		
<code>\tl_count:n</code>	171		
<code>\tl_expandable_lowercase:n</code>	751		
<code>\tl_if_blank:fF</code>	49		
<code>\tl_if_blank:fTF</code>	63		
<code>\tl_if_blank:nF</code>	59		
<code>\tl_if_blank:nTF</code>	67, 70		
<code>\tl_if_empty:NT</code>	796		
<code>\tl_if_eq:NNF</code>	636		
<code>\tl_if_exist:cF</code>	464		
<code>\tl_new:N</code>	677, 678, 785		
<code>\tl_put_right:Nx</code>	477, 487, 496, 502, 511		
<code>\tl_set:cn</code>	466, 570		
<code>\tl_set:cx</code>	556, 563, 590, 624		

\tl_set:Nn 472
 \tl_set:Nx 462,
 598, 600, 602, 604, 606, 608, 610, 705, 706, 732, 733, 750
 \tl_set_eq:NN 644
 \tl_to_lowercase:n 428
 \tl_to_str:N 706, 735
 \tl_to_str:n 737
 tmpa commands:
 \l_tmpa_int 458, 461, 463
 \l_tmpa_tl 462, 464, 466, 472,
 477, 487, 496, 502, 511, 517, 547, 548, 552, 559, 566, 574
 tmpb commands:
 \l_tmpb_tl 554, 559
 token commands:
 \token_if_int_register:NTF 92
 \token_to_meaning:N 690
 \token_to_str:N 689
 \TrimSpaces 456
 true commands:
 \c_true_bool 180, 207, 213, 214, 694
 twelve commands:
 \c_twelve 391
 two commands:
 \c_two 176, 248, 264

U

use commands:
 \use:c 445
 \use:n 114, 122, 134, 146, 303, 310, 327
 \use:x 514
 \use_i:nn 165, 692, 695
 \use_ii:nn 167, 699
 \use_none:n 193, 268
 \use_none:nn 194
 \use_none:nnn 195, 659

Z

zero commands:
 \c_zero 98, 101, 116, 154, 182,
 186, 206, 212, 230, 241, 244, 245, 252, 253, 258, 273, 338
 \zhcurrtime 2, 12, 437
 \zhdate 2, 10, 346
 \zhdigits 1, 3, 8, 275
 \zhdigitsoptions 8, 281, 282, 286
 \zhnum 1, 3, 5, 75
 zhnum commands:
 \l_zhnum_active_char_bool 711, 736, 742, 743, 783
 \l_zhnum_ancient_bool 248, 263, 761, 766, 771
 \c_zhnum_and_tl 52
 \zhnum_assgin_const_tl:cx 576, 584, 616, 622, 624
 \zhnum_blank_to_zero:f 44, 46, 54, 56, 62
 \zhnum_blank_to_zero:n 5, 68, 74
 \l_zhnum_byte_max_int 719, 722, 726, 729
 \l_zhnum_byte_min_int 716, 717, 721, 725, 728
 \l_zhnum_cfg_map_finan_prop
 13, 527, 530, 535, 547, 574, 668
 \l_zhnum_cfg_map_prop . 13, 520, 523, 533, 538, 540, 666
 \l_zhnum_cfg_map_var_prop
 13, 524, 531, 534, 552, 554, 667
 \l_zhnum_cfg_tl .. 635, 636, 643, 644, 666, 667, 668, 677
 \zhnum_check_financial:nn 13, 540, 572
 \zhnum_check_simp:nn 13, 538, 544
 _zhnum_check_simp_aux:nn 546, 548, 550

\zhnum_check_time:Nn
 .. 10, 354, 355, 356, 360, 361, 362, 364, 433, 434, 439, 441
 \zhnum_counter:n 5, 78, 85, 88
 \zhnum_counter:Nn 89, 90
 _zhnum_date:www 348, 352
 \c_zhnum_day_tl 356, 362
 \l_zhnum_day_tl 611
 \zhnum_decimal:nn 4, 32, 60
 \zhnum_digit_map:n
 12, 177, 243, 249, 250, 268, 269, 273, 444
 \zhnum_digits:Nn 9, 278, 290, 294, 296, 298
 \zhnum_digits_null:n 9, 295, 297, 354
 \zhnum_digits_null:V 360
 \zhnum_digits_zero:n 9, 65, 293
 \c_zhnum_dot_tl 62, 324
 \l_zhnum_encoding_tl
 705, 706, 714, 732, 735, 741, 750, 752, 785, 796
 _zhnum_fraction:www 4, 38, 40
 \c_zhnum_fri_tl 377
 \l_zhnum_fri_tl 606
 \c_zhnum_hour_tl 433, 440
 \c_zhnum_hundred_tl 250, 629
 \zhnum_if_digit:NTF 6, 142, 162, 320
 \zhnum_if_unicode_engine:TF 16, 695, 699, 798
 \zhnum_if_unicode_engine_p: 16, 694, 698, 701
 \zhnum_input_cfg:n 645, 648
 \zhnum_insert_zeros:n 7, 179, 190
 \zhnum_int:n 93, 96, 361, 362, 439, 441
 \zhnum_integer:n 5, 9, 37, 109, 355, 356, 433, 434
 _zhnum_integer_or_fraction:www 4, 31, 34
 \l_zhnum_last_cfg_tl 636, 644, 678
 \zhnum_load_cfg:n 15, 633, 640
 \zhnum_load_cfg:o 741, 752
 _zhnum_loop_end:wnn 144, 150
 \c_zhnum_minus_tl 103, 158, 332
 \l_zhnum_minus_tl 474
 \c_zhnum_minute_tl 434, 442
 \c_zhnum_mon_tl 373
 \l_zhnum_mon_tl 598
 \c_zhnum_month_tl 355, 361
 \l_zhnum_normal_bool 578, 762, 767, 772
 \l_zhnum_null_bool 592, 777
 \l_zhnum_null_tl 475, 593
 \zhnum_number:n 4, 16, 23, 26, 51, 72
 _zhnum_number:www 4, 27, 28
 _zhnum_number_item:fNNNN 232
 \zhnum_number_item:nn 8, 201, 226
 _zhnum_number_item:nNNNN 227, 228, 234
 _zhnum_output:nnwn 143, 148
 _zhnum_output_digits:NN 321, 334
 \zhnum_parse_config: 13, 536, 638, 708
 \zhnum_parse_number:f 99, 104
 \zhnum_parse_number:n 7, 170, 189
 \zhnum_parse_number:nn 7, 159, 171, 172
 _zhnum_parse_number:nnn 173, 174
 \c_zhnum_parts_tl 45, 55
 \zhnum_process_number:NNn 8, 210, 237
 \zhnum_process_number:NNNNN 8, 238, 239
 _zhnum_prop_gset_eq:Nn 661, 675
 _zhnum_prop_initial:Nn 653, 670
 _zhnum_read_abs_loop:Nw 6, 136, 140, 145
 _zhnum_read_digits:w 300, 329

_zhnum_read_digits_loop:NN	313, 318, 326	_zhnum_split_number:nnNNnn	204, 224
_zhnum_read_integer:www	6, 111, 152	_zhnum_split_number_aux:NNfnnn	207, 213, 214
_zhnum_read_sign_loop:N	113, 118, 121	_zhnum_split_number_aux:NNnnnn	218, 225
_zhnum_read_sign_loop:NN	302, 306, 309	\c_zhnum_sun_tl	372
_zhnum_read_zeros_loop:N	126, 130, 133	\l_zhnum_sun_tl	610
_zhnum_recursion_stop:NNNNw	231, 235	\c_zhnum_ten_tl	271, 628
\l_zhnum_reset_bool	539, 612, 619, 652, 679	\c_zhnum_thousand_tl	243, 630
\zhnum_reset_config:	16, 708, 740, 782	\c_zhnum_thu_tl	376
_zhnum_result:nn	116, 148, 149, 150	\l_zhnum_thu_tl	604
\c_zhnum_sat_tl	371	_zhnum_time:ww	425, 431
\l_zhnum_sat_tl	608	\l_zhnum_time_bool	365, 779, 780
\g_zhnum_scale_int	452, 453, 454, 465	\c_zhnum_tue_tl	374
\zhnum_scale_map:n	12, 211, 446, 452	\l_zhnum_tue_tl	600
\zhnum_scale_map_hook:n	449, 455, 469	\zhnum_two_digits:n	10, 383, 394
\zhnum_scale_map_loop:n	12, 451, 455	\zhnum_update_cfg:n	637, 641
\zhnum_set_active:	711, 712	_zhnum_update_cfg_prop:N	646, 653, 661, 664
\zhnum_set_alias:	15, 625, 632	\c_zhnum_wed_tl	375
\zhnum_set_catcode:	16, 655, 702, 710	\l_zhnum_wed_tl	602
\zhnum_set_cfg_name:Nn	16, 635, 703, 730	_zhnum_week_day:www	10, 350, 367, 368
\zhnum_set_digits_map:nn	13, 519, 809, 811, 813, 816, 818, 819, 820, 821, 822, 823, 824, 825, 826, 827, 828, 829, 830, 831, 832, 833, 839, 841, 842, 843, 851, 852, 854, 855, 856, 857, 858, 866, 867, 869, 875, 877, 878, 879, 885, 887, 888	\c_zhnum_weekday_tl	599, 601, 603, 605, 607, 609, 611
\zhnum_set_digits_map:nnn	13, 521, 805, 806, 835, 836, 845, 846, 847, 848, 860, 861, 862, 863, 871, 872, 881, 882	\c_zhnum_year_tl	354, 360
\zhnum_set_financial_map:nn	13, 526, 889, 890, 891, 892, 898, 900, 901, 907, 909, 910, 911, 912, 913, 914	\zhnum_Zeller:nnn	10, 370, 380
\zhnum_set_financial_map:nnn	13, 528, 894, 895, 903, 904	_zhnum_Zeller_aux:Nnnn	384, 385, 388
\zhnum_set_week_day:	13, 542, 596	\zhnum_Zeller_aux:Nnnn	10
\zhnum_set_zero:	13, 541, 588	\zhnum_Zeller_Gregorian:nnn	11, 384, 399
\l_zhnum_simp_bool	558, 565, 774, 775	\zhnum_Zeller_Julian:nnn	11, 385, 412
\zhnum_split_number:fnNfn	179	\c_zhnum_zero_tl	64, 71, 106, 155, 209, 242, 245, 253, 627
_zhnum_split_number:fnNNnn	201	\zhnumber	1, 3, 3, 13
\zhnum_split_number:nNNnn	7, 199, 221, 223	\zhnumberwithoptions	4, 5, 17, 19
		\zhnumExtendScaleMap	2, 12, 456
		\zhnumsetup	2, 17, 789
		\zhnumwithoptions	79, 81
		\zhtime	2, 11, 424
		\zhtoday	2, 10, 358
		\zhweekday	2, 10, 366