zhnumber 宏包

李清
sobenlee@gmail.com

2022/07/14　　v3.0*

第 1 节　简介

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

zhnumber 支持 GBK，Big5 和 UTF8 编码，依赖 LaTeX3 项目的 expl3，xparse 和 l3keys2e 宏包。

第 2 节　使用方法

encoding

encoding = ⟨*GBK*|*Big5*|**UTF8**⟩

Updated: 2022-07-10

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、XeLaTeX 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，默认也使用 UTF8。

\zhnumber ⋆

\zhnumber {⟨*number*⟩}

Updated: 2022-07-10

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

```
\zhnumber{2012020120}\\
\zhnumber{2 012 020 120}\\
\zhnumber{2,012,020,120}\\
\zhnumber{2012.020120}\\
\zhnumber{2012.}\\
\zhnumber{.2012}\\
\zhnumber{20120/20120}\\
\zhnumber{/2012}\\
\zhnumber{2012/}\\
\zhnumber{201;2020/120}
```

\zhdigits ⋆

\zhdigits　{⟨*number*⟩}
\zhdigits * {⟨*number*⟩}

Updated: 2022-07-10

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

二〇一二〇二〇一二〇
二零一二零二零一二零

```
\zhdigits{2012020120}\\
\zhdigits*{2012020120}
```

*ctex-kit rev. 5e8c3fe.

\zhnum ⋆ \zhnum {⟨*counter*⟩}
\pagenumbering {zhnum}
Updated: 2022-07-10
与 \roman 等类似，用于将 LaTeX 计数器的值转换为中文数字。例如

二

```
\zhnum{section}
```

\zhdig ⋆ \zhdig {⟨*counter*⟩}
\pagenumbering {zhdig}
Updated: 2022-07-10
与 \roman 等类似，用于将 LaTeX 计数器的值转换为中文数字串。例如

二

```
\zhdig{section}
```

\zhweekday ⋆ \zhweekday {⟨*yyyy/mm/dd*⟩}
Updated: 2022-07-10
输出日期当天的星期。例如

星期日

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

\zhdate ⋆ \zhdate {⟨*yyyy/mm/dd*⟩}
\zhdate * {⟨*yyyy/mm/dd*⟩}
Updated: 2022-07-10
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

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

\zhtoday ⋆
Updated: 2022-07-10
与 \today 类似，以中文输出当天的日期。例如

2022 年 7 月 14 日

```
\zhtoday
```

\zhtime ⋆ \zhtime {⟨*hh:mm*⟩}
Updated: 2022-07-10
以中文格式输出时间。例如

23 时 56 分

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

\zhcurrtime ⋆
Updated: 2022-07-10
输出当前的时间。例如

18 时 55 分

```
\zhcurrtime
```

\zhtiangan ⋆ \zhtiangan {⟨*number*⟩}
Updated: 2022-07-10
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

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

\zhdizhi ⋆ \zhdizhi {⟨*number*⟩}
Updated: 2022-07-10
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

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

\zhganzhi ⋆ \zhganzhi {⟨*number*⟩}
Updated: 2022-07-10
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

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

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

Updated: 2022-07-10 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 壬寅

```
\zhganzhinian{1898}  \zhganzhinian{-246} \\
\zhganzhinian{-2697} \zhganzhinian{\year}
```

\zhnumExtendScaleMap `\zhnumExtendScaleMap [⟨character⟩] {⟨character$_1$⟩, ⟨character$_2$⟩, ..., ⟨character$_n$⟩}`

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*$_i$⟩ 设置 $10^{4(i+11)}$。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。

\zhnumsetup `\zhnumsetup {⟨key$_1$⟩=⟨val$_1$⟩, ⟨key$_2$⟩=⟨val$_2$⟩, ...}`

Updated: 2022-07-10 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。以**粗体**表示选项的默认值。

time `time = ⟨`**`Arabic`**`|`*Chinese*`⟩`

New: 2012-05-25 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。例如

二〇二二年七月十四日十八时五十五分

```
\zhnumsetup{time=Chinese}
\zhtoday\zhcurrtime
```

arabicsep `arabicsep = {⟨sep⟩}`

New: 2016-05-01 设置日期和时间的数字格式为阿拉伯数字时，阿拉伯数字与汉字的间隔内容。默认为一个空格。

style `style = ⟨`**`Simplified`**`|`*Traditional*`|`**`Normal`**`|`*Financial*`|`*Ancient*`⟩`

Updated: 2012-05-25 意义分别为

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

可以设置 style 为其中一个，也可以是前两个与后三个的适当组合，默认是简体小写。例如

陆萬貳仟零壹拾貳點叁
廿一

```
\zhnumsetup{style={Traditional,Financial}}
\zhnumber{62012.3}\\
\zhnumsetup{style=Ancient}
\zhnumber{21}
```

null `null = ⟨`*true*`|`**`false`**`⟩`

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

ganzhi-cyclic　ganzhi-cyclic = ⟨*true*|**false**⟩

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

```
\zhnumsetup{ganzhi-cyclic}
\zhtiangan{11} \zhtiangan{12} \zhtiangan{209}
\zhtiangan{-1} \zhtiangan{-2} \zhtiangan{-683} \\
\zhdizhi{13}   \zhdizhi{24}   \zhdizhi{1211}
\zhdizhi{-1}   \zhdizhi{-2}   \zhdizhi{-8199}  \\
\zhganzhi{61}  \zhganzhi{72}  \zhganzhi{2158}  \\
\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。其他的选项同字面意思，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　reset

Updated: 2022-07-10　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　activechar = ⟨**true**|*false*⟩

New: 2014-09-09　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　\zhnumber　[⟨*options*⟩] {⟨*number*⟩}
\zhdigits　\zhdigits *　[⟨*options*⟩] {⟨*number*⟩}
\zhnum　\zhnum　[⟨*options*⟩] {⟨*counter*⟩}
\zhdig　\zhdig　[⟨*options*⟩] {⟨*counter*⟩}

Updated: 2022-07-10　如果只改变当前数字的中文输出格式，可以使用带选项的格式转换命令，其中 ⟨*options*⟩ 与 \zhnumsetup 的参数相同，如上所介绍。这些带了选项的命令是不可展开的，在某些场合使用时要小心。

第 3 节　zhnumber 宏包代码实现

```
⟨*package⟩
⟨@@=zhnum⟩
\msg_new:nnn { zhnumber } { l3-too-old }
```

```
  {
    Support~package~'expl3'~too~old. \\\\
    Please~update~an~up~to~date~version~of~the~bundles\\\\
    'l3kernel'~and~'l3packages'\\\\
    using~your~TeX~package~manager~or~from~CTAN.
  }
\@ifpackagelater { expl3 } { 2019/03/05 } { }
  { \msg_error:nn  { zhnumber }  { l3-too-old } }
\cs_if_exist:NF \NewDocumentCommand
  { \RequirePackage { xparse } }
```

\zhnum_output:n
\zhnum_expand_wrap:wn

受益于 \tex_expanded:D，我们使用如下的结构简单实现 f 展开。

```
\exp_not:e
  {
    ...
    \exp_not:o { <tl_var> }
    ...
    \exp_not:o { <tl_var> }
    ...
    \exp_not:o { <tl_var> }
    ...
  }
```

并且可以将 \zhnumber 等无风险地使用于诸如 \edef 定义等完全展开的场合。从 TeX Live 2019 开始，各个主要引擎都已经支持 \tex_expanded:D，对于较早的版本，LaTeX3 也实现了一个模拟。

```
\cs_new:Npn \zhnum_output:n #1
  { \exp_args:Nc \zhnum_exp_not:o { l__zhnum_ #1 _tl } }
\cs_new_protected:Npn \zhnum_expand_wrap:wn #1#
  { \__zhnum_expand_wrap_aux:nn {#1} }
\cs_new_protected:Npn \__zhnum_expand_wrap_aux:nn #1#2
  { #1 { \exp_not:e {#2} } }
\cs_new_eq:NN \zhnum_exp_not:e \exp_not:e
\cs_new_eq:NN \zhnum_exp_not:o \exp_not:o
```

但这种方式也有一定的缺点，比如在 pdfTeX 引擎下，hyperref 包的 \pdfstringdef 就需要把汉字首字节的活动字符展开，并作适当的重定义后才能得到预期的结果。为此，我们定义一个清除命令，不使用以上机制。

```
\cs_new_protected:Npn \zhnumClearWrapper
  {
    \cs_set_eq:NN \zhnum_exp_not:e \use:n
    \cs_set_eq:NN \zhnum_exp_not:o \use:n
  }
\cs_new_protected:Npn \zhnumResetWrapper
  {
    \cs_set_eq:NN \zhnum_exp_not:e \exp_not:e
    \cs_set_eq:NN \zhnum_exp_not:o \exp_not:o
  }
\hook_gput_code:nnn { package/CJKutf8/after } { zhnumber }
  { \g@addto@macro \pdfstringdefPreHook { \zhnumClearWrapper } }
\hook_gput_code:nnn { package/xCJK2uni/after } { zhnumber }
  { \g@addto@macro \pdfstringdefPreHook { \zhnumClearWrapper } }
```

\zhnumber 用于将输入的数字按照中文格式输出。

```
\NewExpandableDocumentCommand \zhnumber { +o +m }
  {
    \tl_if_novalue:nTF {#1}
      { \zhnum_number:e }
      { \zhnumberwithoptions {#1} }
    {#2}
  }
```

\zhnumberwithoptions 带选项的用户函数。

```
\NewDocumentCommand \zhnumberwithoptions { +m +m }
  {
    \group_begin:
      \zhnum_set:n {#1}
      \zhnum_number:e {#2}
    \group_end:
  }
```

\zhnum_number:n
__zhnum_number:www
先判断输入的是小数还是分数。

```
\zhnum_expand_wrap:wn
\cs_new:Npn \zhnum_number:n #1
  { \__zhnum_number:www #1 . \q_nil . \q_stop }
\cs_new:Npn \__zhnum_number:www #1 . #2 . #3 \q_stop
  {
    \quark_if_nil:nTF {#2}
      { \__zhnum_integer_or_fraction:www #1 / \q_nil / \q_stop }
      { \zhnum_decimal:nn {#1} {#2} }
  }
\cs_generate_variant:Nn \zhnum_number:n { e }
```

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

```
\cs_new:Npn \__zhnum_integer_or_fraction:www #1 / #2 / #3 \q_stop
  {
    \quark_if_nil:nTF {#2}
      { \zhnum_integer:n {#1} }
      { \__zhnum_fraction:wwww #2 \q_mark #1 ; \q_nil ; \q_stop }
  }
```

__zhnum_fraction:wwww 对分数进行预处理。

```
\cs_new:Npn \__zhnum_fraction:wwww #1 \q_mark #2 ; #3 ; #4 \q_stop
  {
    \quark_if_nil:nTF {#3}
      {
        \zhnum_blank_to_zero:n {#1}
        \zhnum_output:n { parts }
        \zhnum_blank_to_zero:n {#2}
      }
      {
        \tl_if_blank:nF {#2}
          {
            \zhnum_number:n {#2}
            \zhnum_output:n { and }
          }
        \zhnum_blank_to_zero:n {#1}
        \zhnum_output:n { parts }
        \zhnum_blank_to_zero:n {#3}
      }
  }
```

\zhnum_decimal:nn 对小数进行预处理。

```
\cs_new:Npn \zhnum_decimal:nn #1#2
  {
    \zhnum_blank_to_zero:n {#1}
    \zhnum_output:n { dot }
    \tl_if_blank:nTF {#2}
      { \zhnum_output:n { 0 } }
      { \zhnum_digits_zero:n {#2} }
  }
```

\zhnum_blank_to_zero:n 输出小数的整数位。

```
\cs_new:Npn \zhnum_blank_to_zero:n #1
  {
```

```
    \tl_if_blank:nTF {#1}
      { \zhnum_output:n { 0 } }
      { \zhnum_number:n {#1} }
  }
```

\zhnum
\zhnumberwithoptions
用于将 LaTeX 计数器按中文格式输出。

```
\NewExpandableDocumentCommand \zhnum { +o +m }
  {
    \tl_if_novalue:nTF {#1}
      { \zhnum_counter:n }
      { \zhnumwithoptions {#1} }
    {#2}
  }
\NewDocumentCommand \zhnumwithoptions { +m +m }
  {
    \group_begin:
      \zhnum_set:n {#1}
      \zhnum_counter:n {#2}
    \group_end:
  }
```

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

```
\cs_new:Npn \zhnum_counter:n #1
  {
    \int_if_exist:cTF { c@#1 }
      { \exp_args:Nc \zhnum_int:n { c@#1 } }
      { \__zhnum_counter_error:n {#1} }
  }
\cs_new:Npn \__zhnum_counter_error:n #1
  { \msg_expandable_error:nnn { zhnumber } { not-counter } {#1} }
\msg_new:nnn  { zhnumber } { not-counter }
  { `#1'~is~not~a~LaTeX~counter. }
\zhnum_expand_wrap:wn
\cs_new:Npn \zhnum_int:n #1
  {
    \int_compare:nNnTF {#1} > \c_zero_int
      { \zhnum_parse_number:f { \int_eval:n {#1} } }
      {
        \int_compare:nNnTF {#1} < \c_zero_int
          {
            \zhnum_output:n { minus }
            \zhnum_parse_number:f { \int_eval:n { - #1 } }
          }
          { \zhnum_output:n { 0 } }
      }
  }
```

\@zhnum
用于支持 \pagenumbering{zhnum}。

```
\cs_new:Npn \@zhnum { \zhnum_int:n }
```

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

```
\cs_new:Npn \zhnum_integer:n #1
  {
    \exp_after:wN \__zhnum_read_integer:www
    \int_value:w
      \exp_after:wN \__zhnum_read_sign_loop:N
      \exp:w \exp_end_continue_f:w \use:n
      #1 \exp_stop_f: \q_recursion_tail \q_recursion_stop
         \__zhnum_result:nn { \c_zero_int } { } ;
  }
\cs_new:Npn \__zhnum_read_sign_loop:N #1
  {
```

```
    \if:w + \if:w - \exp_not:N #1 + \fi: \exp_not:N #1
      \exp_after:wN \__zhnum_read_sign_loop:N
      \exp:w \exp_end_continue_f:w \exp_after:wN \use:n
    \else:
      1 \exp_after:wN ;
      \exp:w \exp_end_continue_f:w
        \exp_after:wN \__zhnum_read_zeros_loop:N
        \exp_after:wN #1
    \fi:
  }
\cs_new:Npn \__zhnum_read_zeros_loop:N #1
  {
    \if:w 0 \exp_not:N #1
      \exp_after:wN \__zhnum_read_zeros_loop:N
      \exp:w \exp_end_continue_f:w \exp_after:wN \use:n
    \else:
      \exp_after:wN \__zhnum_read_abs_loop:Nw
      \exp_after:wN #1
    \fi:
  }
```

__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。

```
\cs_new:Npn \__zhnum_read_abs_loop:Nw #1#2 \q_recursion_stop
  {
    \zhnum_if_digit:NTF #1
      { \__zhnum_output:nnwnn { + 1 } #1 }
      { \quark_if_recursion_tail_stop_do:Nn #1 { \__zhnum_loop_end:wnn } }
    \exp_after:wN \__zhnum_read_abs_loop:Nw
      \exp:w \exp_end_continue_f:w \use:n #2 \q_recursion_stop
  }
\cs_new:Npn \__zhnum_output:nnwnn #1#2#3 \__zhnum_result:nn #4#5
  { #3 \__zhnum_result:nn { #4#1 } { #5#2 } }
\cs_new:Npn \__zhnum_loop_end:wnn #1 \__zhnum_result:nn #2#3
  { \int_eval:n {#2} ; #3 }
```

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

```
\cs_new:Npn \__zhnum_read_integer:www #1 ; #2 ; #3 ;
  {
    \int_compare:nNnTF {#2} = \c_zero_int
      { \zhnum_output:n { 0 } }
      {
        \int_compare:nNnF {#1} = \c_one_int
          { \zhnum_output:n { minus } }
        \zhnum_parse_number:nn {#2} {#3}
      }
  }
```

\zhnum_if_digit:NTF 判断 #1 是否为数字位。

```
\cs_new:Npn \zhnum_if_digit:NTF #1
  {
    \if_int_compare:w 9 < 1 \exp_not:N #1 \exp_stop_f:
      \exp_after:wN \use_i:nn
    \else:
      \exp_after:wN \use_ii:nn
    \fi:
  }
```

\zhnum_parse_number:n
\zhnum_parse_number:nn

```
\cs_new:Npn \zhnum_parse_number:n #1
  { \exp_args:Nf \zhnum_parse_number:nn { \tl_count:n {#1} } {#1} }
\cs_new:Npn \zhnum_parse_number:nn #1
  { \exp_args:Nf \__zhnum_parse_number:nnn { \int_mod:nn {#1} { 4 } } {#1} }
\cs_new:Npn \__zhnum_parse_number:nnn #1#2
  {
```

```
    \int_compare:nNnTF {#2} < 2
      { \zhnum_output:n }
      {
        \int_compare:nNnTF {#1} = \c_zero_int
          { \zhnum_split_number:fn { \int_eval:n { #2 / 4 - 1 } } }
          { \__zhnum_split_number_aux:nnn {#1} {#2} }
      }
  }
\cs_generate_variant:Nn \zhnum_parse_number:n { f }
```

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

```
\cs_new:Npn \__zhnum_split_number_aux:nnn #1#2
  {
    \exp_after:wN \__zhnum_split_number_aux:wwn
      \int_value:w \int_div_truncate:nn {#2} { 4 }
        \if_case:w #1 \exp_stop_f:
          \or: \exp_after:wN \use:n
          \or: \exp_after:wN \use_i_ii:nnn
          \or: \exp_after:wN \use_i:nnn
        \fi:
        { \exp_stop_f: ; 0 } 0 0 ;
  }
\cs_new:Npn \__zhnum_split_number_aux:wwn #1 ; #2 ; #3
  { \zhnum_split_number:nn {#1} { #2#3 } }
```

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

```
\cs_new:Npn \zhnum_split_number:nn #1#2
  {
    \zhnum_split_number:NNnNNNNw \c_true_bool \c_true_bool {#1}
      #2 \q_recursion_tail \q_nil \q_nil \q_nil \q_recursion_stop
  }
\cs_generate_variant:Nn \zhnum_split_number:nn { f }
```

\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。

```
\cs_new:Npn \zhnum_split_number:NNnNNNNw #1#2#3#4#5#6#7
  {
    \quark_if_recursion_tail_stop:N #4
    \int_compare:nNnTF { #4#5#6#7 } = \c_zero_int
      { \use_i:nn }
      {
        \bool_if:NF #1 { \zhnum_output:n { 0 } }
        \zhnum_process_number:NNNNNN #4#5#6#7#1#2
        \zhnum_scale_map:n {#3}
        \int_compare:nNnTF {#7} = \c_zero_int
      }
      { \zhnum_split_number:NNfNNNNw \c_false_bool \c_true_bool }
      { \zhnum_split_number:NNfNNNNw \c_true_bool  \c_false_bool }
    { \int_eval:n { #3 - 1 } }
  }
\cs_generate_variant:Nn \zhnum_split_number:NNnNNNNw { NNf }
```

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

```
\cs_new:Npn \zhnum_process_number:NNNNNN #1#2#3#4#5#6
  {
    \int_compare:nNnTF {#1} = \c_zero_int
      {
        \bool_if:NF #6
          { \zhnum_output:n { 0 } }
      }
      {
        \zhnum_output:n {#1}
        \zhnum_output:n { 1000 }
      }
    \int_compare:nNnTF {#2} = \c_zero_int
      {
```

```
      \int_compare:nNnF { #1 * (#3#4) } = \c_zero_int
        { \zhnum_output:n { 0 } }
    }
    {
      \int_compare:nNnTF {#2} = 2
        { \zhnum_output:n { 200 } }
        {
          \zhnum_output:n {#2}
          \zhnum_output:n { 100 }
        }
    }
  \int_compare:nNnTF {#3} = \c_zero_int
    {
      \int_compare:nNnF { #2 * #4 } = \c_zero_int
        { \zhnum_output:n { 0 } }
    }
    {
      \bool_lazy_all:nTF
        {
          { \int_compare_p:nNn {#3}   = \c_one_int }
          { \int_compare_p:nNn {#1#2} = \c_zero_int }
          {#6}
          {#5}
        }
        { \zhnum_output:n { 10 } }
        {
          \int_compare:nTF { 1 < #3 < 5 }
            { \zhnum_output:n { #3 0 } }
            {
              \zhnum_output:n {#3}
              \zhnum_output:n { 10 }
            }
        }
    }
  \int_compare:nNnF {#4} = \c_zero_int
    { \zhnum_output:n {#4} }
}
```

\zhdig 用于将 LaTeX 计数器按中文数字串输出。

```
\NewExpandableDocumentCommand \zhdig { +o +m }
  {
    \tl_if_novalue:nTF {#1}
      { \zhnum_digits_counter:n }
      { \zhdigwithoptions {#1} }
    {#2}
  }
\NewDocumentCommand \zhdigwithoptions { +m +m }
  {
    \group_begin:
      \zhnum_set:n {#1}
      \zhnum_digits_counter:n #1 {#2}
    \group_end:
  }
\cs_new:Npn \zhnum_digits_counter:n #1
  {
    \int_if_exist:cTF { c@#1 }
      { \zhnum_digits_null:v { c@#1 } }
      { \__zhnum_counter_error:n {#1} }
  }
```

\@zhdig 用于支持 \pagenumbering{zhdig}。

```
\cs_new:Npn \@zhdig #1
  { \zhnum_digits_null:f { \int_eval:n {#1} } }
```

\zhdigits 将输入的数字输出为中文数字串输出。

\zhdigitswithoptions

```
\NewExpandableDocumentCommand \zhdigits { +s +o +m }
```

```
  {
    \tl_if_novalue:nTF {#2}
      { \zhnum_digits:Ne #1 }
      { \zhdigitswithoptions {#1} {#2} }
    {#3}
  }
\NewDocumentCommand \zhdigitswithoptions { +m +m +m }
  {
    \group_begin:
      \zhnum_set:n {#2}
      \zhnum_digits:Ne #1 {#3}
    \group_end:
  }
```

\zhnum_digits_zero:n
\zhnum_digits_null:n

快捷方式。

```
\cs_new:Npn \zhnum_digits_zero:n
  { \zhnum_digits:Nn \c_true_bool }
\cs_new:Npn \zhnum_digits_null:n
  { \zhnum_digits:Nn \c_false_bool }
\cs_generate_variant:Nn \zhnum_digits_null:n { v , f }
```

\zhnum_digits:Nn

与 \zhnum_integer:n 类似，但不用去掉多余的零。

```
\zhnum_expand_wrap:wn
\cs_new:Npn \zhnum_digits:Nn #1#2
  {
    \exp_after:wN \__zhnum_read_digits:w
    \int_value:w
      \exp_after:wN \__zhnum_read_sign_loop:NN \exp_after:wN #1
      \exp:w \exp_end_continue_f:w \use:n
      #2 \exp_stop_f: \q_recursion_tail \q_recursion_stop
  }
\cs_new:Npn \__zhnum_read_sign_loop:NN #1#2
  {
    \if:w + \if:w - \exp_not:N #2 + \fi: \exp_not:N #2
      \exp_after:wN \__zhnum_read_sign_loop:NN \exp_after:wN #1
      \exp:w \exp_end_continue_f:w \exp_after:wN \use:n
    \else:
      1 \exp_after:wN ;
        \exp_after:wN \__zhnum_read_digits_loop:NN
        \exp_after:wN #1
        \exp_after:wN #2
    \fi:
  }
\cs_new:Npn \__zhnum_read_digits_loop:NN #1#2
  {
    \zhnum_if_digit:NTF #2
      { \__zhnum_output_digits:NN #1#2 }
      {
        \quark_if_recursion_tail_stop:N #2
        \token_if_eq_charcode:NNT #2 .
          { \zhnum_output:n { dot } }
      }
    \exp_after:wN \__zhnum_read_digits_loop:NN \exp_after:wN #1
      \exp:w \exp_end_continue_f:w \use:n
  }
\cs_new:Npn \__zhnum_read_digits:w #1 ;
  {
    \int_compare:nNnF {#1} = \c_one_int
      { \zhnum_output:n { minus } }
  }
\cs_new:Npn \__zhnum_output_digits:NN #1#2
  {
    \zhnum_output:n
      {
        \if_int_compare:w #2 = \c_zero_int
          \bool_if:NTF #1 { zero } { null }
```

```
        \else:
          #2
        \fi:
      }
  }
\cs_generate_variant:Nn \zhnum_digits:Nn { Ne }
```

\zhdate 输出中文日期。

```
\NewExpandableDocumentCommand \zhdate { +s +m }
  {
    \__zhnum_date:www #2 \q_stop
    \bool_if:NT #1 { \__zhnum_week_day:www #2 \q_stop }
  }
\cs_new:Npn \__zhnum_date:www #1/#2/#3 \q_stop
  { \__zhnum_date_aux:nnn {#1} {#2} {#3} }
```

\zhtoday 输出当天日期。

```
\cs_new:Npn \zhtoday
  {
    \__zhnum_date_aux:nnn
      { \int_value:w \tex_year:D }
      { \tex_month:D }
      { \tex_day:D }
  }
```

__zhnum_date_aux:nnn

```
\cs_new:Npn \__zhnum_date_aux:nnn
  {
    \bool_if:NTF \l__zhnum_time_bool
      { \__zhnum_date_aux:NNnnnn \zhnum_digits_null:n \zhnum_int:n { } }
      { \__zhnum_date_aux:Nnnnn \int_to_arabic:n { \l__zhnum_arabic_sep_tl } }
  }
\cs_new:Npn \__zhnum_date_aux:Nnnnn #1#2
  { \__zhnum_date_aux:NNnnnn #1#1 { \zhnum_exp_not:o {#2} } }
\zhnum_expand_wrap:wn
\cs_new:Npn \__zhnum_date_aux:NNnnnn #1#2#3#4#5#6
  {
    #1 {#4} #3 \zhnum_output:n { year }  #3
    #2 {#5} #3 \zhnum_output:n { month } #3
    #2 {#6} #3 \zhnum_output:n { day }
  }
```

\zhweekday 输出星期

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

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

```
\zhnum_expand_wrap:wn
\cs_new:Npn \__zhnum_week_day:www #1/#2/#3 \q_stop
  {
    \if_case:w \zhnum_Zeller:nnn {#1} {#2} {#3} \exp_stop_f:
          \zhnum_output:n { sat }
      \or: \zhnum_output:n { sun }
      \or: \zhnum_output:n { mon }
      \or: \zhnum_output:n { tue }
      \or: \zhnum_output:n { wed }
      \or: \zhnum_output:n { thu }
      \or: \zhnum_output:n { fri }
    \fi:
  }
```

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

用 Zeller 公式[1] 计算星期几。

```
\cs_new:Npn \zhnum_Zeller:nnn #1#2#3
```

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

```
  {
    \int_compare:nNnTF
      { #1 \zhnum_two_digits:n {#2} \zhnum_two_digits:n {#3} } > { 1582 10 04 }
      { \__zhnum_Zeller_aux:Nnnn \zhnum_Zeller_Gregorian:nnn }
      { \__zhnum_Zeller_aux:Nnnn \zhnum_Zeller_Julian:nnn }
    {#1} {#2} {#3}
  }
\cs_new:Npn \__zhnum_Zeller_aux:Nnnn  #1#2#3#4
  {
    \int_compare:nNnTF {#3} < 3
      { #1 { #2 - 1 } { #3 + 12 } {#4} }
      { #1 {#2} {#3} {#4} }
  }
\cs_new:Npn \zhnum_two_digits:n #1
  {
    \int_compare:nNnT {#1} < { 10 } { 0 }
    \int_eval:n {#1}
  }
```

\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 $m \mathrel{+}= 12$，同时 $Y --$。

```
\cs_new:Npn \zhnum_Zeller_Gregorian:nnn #1#2#3
  {
    \int_mod:nn
      {
          (#3)
        + \int_div_truncate:nn { 26 * ( #2 + 1 ) } { 10 }
        + (#1)
        + \int_div_truncate:nn {#1} { 4 }
        + 6 * \int_div_truncate:nn {#1} { 100 }
        + \int_div_truncate:nn {#1} { 400 }
      }
      { 7 }
  }
```

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

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

```
\cs_new:Npn \zhnum_Zeller_Julian:nnn #1#2#3
  {
    \int_mod:nn
      {
          (#3)
        + \int_div_truncate:nn { 26 * ( #2 + 1 ) } { 10 }
        + (#1)
        + \int_div_truncate:nn {#1} { 4 }
        + 5
      }
      { 7 }
  }
```

\zhtime 输出时间。

```
\zhnum_expand_wrap:wn
\cs_new:Npn \zhtime #1
  { \__zhnum_time:ww #1 \q_stop }
\use:e
  {
    \cs_new:Npn \exp_not:N \__zhnum_time:ww
      #1 \c_colon_str #2 \exp_not:N \q_stop
  }
  { \__zhnum_time_aux:nn {#1} {#2} }
```

\zhcurrtime 输出当前时间。

```
\zhnum_expand_wrap:wn
\cs_new:Npn \zhcurrtime
  {
    \__zhnum_time_aux:nn
      { \int_div_truncate:nn \tex_time:D { 60 } }
      { \int_mod:nn \tex_time:D { 60 } }
  }
```

__zhnum_time_aux:nn
__zhnum_time_aux:Nnnn

```
\cs_new:Npn \__zhnum_time_aux:nn
  {
    \bool_if:NTF \l__zhnum_time_bool
      { \__zhnum_time_aux:Nnnn \zhnum_int:n { } }
      { \__zhnum_time_aux:Nnnn \int_to_arabic:n { \l__zhnum_arabic_sep_tl } }
  }
\cs_new:Npn \__zhnum_time_aux:Nnnn #1#2#3#4
  {
    #1 {#3} #2 \zhnum_output:n { hour } #2
    #1 {#4} #2 \zhnum_output:n { minute }
  }
```

\zhnum_scale_map:n
\zhnum_scale_map_loop:n
大数系统的映射。

```
\cs_new:Npn \zhnum_scale_map:n #1
  {
    \tl_if_exist:cTF { l__zhnum_s #1 _tl }
      { \zhnum_output:n { s #1 } }
      { \zhnum_scale_map_hook:n {#1} }
  }
\cs_new:Npn \zhnum_scale_map_loop:n #1
  { \zhnum_scale_map:n { \int_mod:nn {#1} \l__zhnum_scale_int } }
\cs_generate_variant:Nn \zhnum_scale_map:n { f }
\int_new:N \l__zhnum_scale_int
\int_set:Nn \l__zhnum_scale_int { 11 }
\cs_new_eq:NN \zhnum_scale_map_hook:n \zhnum_scale_map_loop:n
\tl_new:c { l__zhnum_s 0 _tl }
```

\zhnumExtendScaleMap 扩展进位系统。

```
\NewDocumentCommand \zhnumExtendScaleMap { > { \TrimSpaces } +o +m }
  {
    \int_zero:N \l__zhnum_tmp_int
    \clist_map_function:nN {#2} \zhnum_set_scale:n
    \tl_if_novalue:nF {#1}
      { \cs_set:Npn \zhnum_scale_map_hook:n ##1 {#1} }
  }
```

\zhnum_set_scale:n

```
\cs_new_protected:Npn \zhnum_set_scale:n #1
  {
    \int_incr:N \l__zhnum_tmp_int
    \exp_args:Nc \__zhnum_set_scale:Nn
      { l__zhnum_s \int_eval:n { \l__zhnum_tmp_int + 11 } _tl }
      {#1}
  }
\cs_new_protected:Npn \__zhnum_set_scale:Nn #1
  {
    \tl_if_exist:NF #1
      {
        \tl_new:N #1
        \int_incr:N \l__zhnum_scale_int
      }
    \tl_set:Nn #1
  }
\int_new:N \l__zhnum_tmp_int
```

\zhnum_ganzhi_normal:nnn 保证干支的参数为正数。

```
\zhnum_expand_wrap:wn
```

```
\cs_new:Npn \zhnum_ganzhi_normal:nnn #1#2#3
  {
    \int_compare:nNnF {#1} < \c_one_int
      {
        \cs_if_free:cF { l__zhnum_ #2 _ #1 _tl }
          { \zhnum_output:n { #2 _ #1 } }
      }
  }
```

\zhnum_ganzhi_cyclic:nnn
__zhnum_ganzhi_cyclic_mod:nnnn

对超出范围的数字取模，参数 0 的结果是空值。

```
\zhnum_expand_wrap:wn
\cs_new:Npn \zhnum_ganzhi_cyclic:nnn #1#2#3
  {
    \int_compare:nNnF {#1} = \c_zero_int
      {
        \tl_if_exist:cTF { l__zhnum_ #2 _ #1 _tl }
          { \zhnum_output:n { #2 _ #1 } }
          {
            \__zhnum_ganzhi_cyclic_mod:fnnn
              { \int_mod:nn {#1} {#3} } {#1} {#2} {#3}
          }
      }
  }
\cs_new:Npn \__zhnum_ganzhi_cyclic_mod:nnnn #1#2#3#4
  {
    \int_compare:nNnTF {#2} > \c_zero_int
      { \zhnum_output:n { #3 _ #1 } }
      {
        \int_compare:nNnTF {#1} = \c_zero_int
          { \zhnum_output:n { #3 _ 1 } }
          { \zhnum_output:n { #3 _ \int_eval:n {  #1 + #4 + 1 } } }
      }
  }
\cs_generate_variant:Nn \__zhnum_ganzhi_cyclic_mod:nnnn { f }
```

\zhnum_ganzhi:nnn 默认不对超出范围的数字取模。

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

\zhtiangan 天干。

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

\zhdizhi 地支。

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

\zhganzhi 干支。

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

\zhganzhinian 干支纪年。

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

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

```
\zhnum_expand_wrap:wn
\cs_new:Npn \zhnum_ganzhi_nian:n #1
  {
    \int_compare:nNnTF {#1} > \c_zero_int
      { \zhnum_output:n { ganzhi_ \int_mod:nn { #1 + 57 } { 60 } } }
      {
        \int_compare:nNnF {#1} = \c_zero_int
```

```
          {
            \zhnum_output:n
              {
                ganzhi_ \int_eval:n
                  { \int_mod:nn { #1 - 2 } { 60 } + 60 }
              }
          }
      }
  }
\cs_generate_variant:Nn \zhnum_ganzhi_nian:n { f }
```

根据需要设置中文阿拉伯数字。

```
\tl_new:N \l__zhnum_kv_tl
\tl_new:N \l__zhnum_tmp_tl
\group_begin:
  \tl_build_begin:N \l__zhnum_kv_tl
  \int_step_inline:nn { 10 }
    {
      \tl_new:c { l__zhnum_ #1 _tl }
      \tl_build_put_right:Nx \l__zhnum_kv_tl
        {
            #1 .tl_set:N = \exp_not:c { l__zhnum_normal_    #1 _tl } ,
           F#1 .tl_set:N = \exp_not:c { l__zhnum_financial_ #1 _tl } ,
          E \int_eval:n { #1 * 4 }
               .tl_set:N = \exp_not:c { l__zhnum_ s#1 _tl } ,
        }
    }
  \clist_map_inline:nn { 0 , 100 , 1000 }
    {
      \tl_new:c { l__zhnum_ #1 _tl }
      \tl_build_put_right:Nx \l__zhnum_kv_tl
        {
           #1 .tl_set:N = \exp_not:c { l__zhnum_normal_    #1 _tl } ,
          F#1 .tl_set:N = \exp_not:c { l__zhnum_financial_ #1 _tl } ,
        }
    }
  \clist_map_inline:nn
    {
      20 , 30 , 40 , 200 ,
      dot , and , parts , year , month , day , hour , minute
    }
    {
      \tl_build_put_right:Nx \l__zhnum_kv_tl
        { #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } , }
    }
  \tl_build_put_right:Nx \l__zhnum_kv_tl
    {
      -   .tl_set:N = \exp_not:N \l__zhnum_minus_tl ,
      -0  .tl_set:N = \exp_not:N \l__zhnum_null_tl ,
      E2  .tl_set:N = \exp_not:c { l__zhnum_normal_     100 _tl } ,
      E3  .tl_set:N = \exp_not:c { l__zhnum_normal_    1000 _tl } ,
      FE2 .tl_set:N = \exp_not:c { l__zhnum_financial_  100 _tl } ,
      FE3 .tl_set:N = \exp_not:c { l__zhnum_financial_ 1000 _tl } ,
      E44 .tl_set:N = \exp_not:c { l__zhnum_ s11 _tl }
    }
  \tl_build_get:NN \l__zhnum_kv_tl \l__zhnum_tmp_tl
  \cs_set:Npn \__zhnum_tmp:w #1 . #2 \q_stop
    { , #1 .groups:n = { user } }
  \clist_map_inline:Nn \l__zhnum_tmp_tl
    {
      \tl_build_put_right:Nx \l__zhnum_kv_tl
        { \__zhnum_tmp:w #1 \q_stop }
    }
  \tl_build_put_right:Nn \l__zhnum_kv_tl
    {
      ,
      weekday .tl_set:N = \l__zhnum_weekday_tl ,
```

```
      weekday .groups:n = { user , pre , weekday } ,
    }
  \clist_map_inline:nn
    { mon , tue , wed , thu , fri , sat , sun }
    {
      \tl_build_put_right:Nx \l__zhnum_kv_tl
        {
          #1 .tl_set:N = \exp_not:c { l__zhnum_ #1 _tl } ,
          #1 .groups:n = { user , pos , day } ,
        }
    }
  \int_step_inline:nn { 10 }
    {
      \tl_build_put_right:Nx \l__zhnum_kv_tl
        {
          T#1 .tl_set:N = \exp_not:c { l__zhnum_ganzhi_ #1 _tl } ,
          T#1 .groups:n = { user , pre , tiandi } ,
        }
    }
  \int_step_inline:nn { 12 }
    {
      \tl_build_put_right:Nx \l__zhnum_kv_tl
        {
          D#1 .tl_set:N = \exp_not:c { l__zhnum_dizhi_ #1 _tl } ,
          D#1 .groups:n = { user , pre , tiandi } ,
        }
    }
  \int_step_inline:nn { 60 }
    {
      \tl_build_put_right:Nx \l__zhnum_kv_tl
        {
          GZ#1 .tl_set:N = \exp_not:c { l__zhnum_ganzhi_ #1 _tl } ,
          GZ#1 .groups:n = { user , pos , ganzhi } ,
        }
    }
  \tl_build_end:N \l__zhnum_kv_tl
  \exp_args:NNno \group_end:
\keys_define:nn
  { zhnum / options }
  { \l__zhnum_kv_tl }
```

\zhnum_set_digits_map:nn
\zhnum_set_digits_map:nnn
\zhnum_set_financial_map:nn
\zhnum_set_financial_map:nnn
\zhnum_set_tiangan_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

将配置文件中的中文数字保存到 prop 变量中。

```
\cs_new_protected:Npn \zhnum_set_digits_map:nn #1
  { \prop_put:Nnn \l__zhnum_cfg_map_prop {#1} }
\cs_new_protected:Npn \zhnum_set_digits_map:nnn #1#2#3
  {
    \prop_put_if_new:Nnn \l__zhnum_cfg_map_prop {#1} {#3}
    \prop_put:Nnn \l__zhnum_cfg_map_var_prop {#1_#2} {#3}
  }
\cs_new_protected:Npn \zhnum_set_financial_map:nn #1
  { \prop_put:Nnn \l__zhnum_cfg_map_finan_prop {#1} }
\cs_new_protected:Npn \zhnum_set_financial_map:nnn #1#2#3
  {
    \prop_put_if_new:Nnn \l__zhnum_cfg_map_finan_prop {#1} {#3}
    \prop_put:Nnn \l__zhnum_cfg_map_var_prop { financial_#1_#2 } {#3}
  }
\cs_new_protected:Npn \zhnum_set_tiangan_map:nn #1
  { \prop_put:Nnn \l__zhnum_cfg_map_ganzhi_prop { tiangan_#1 } }
\cs_new_protected:Npn \zhnum_set_dizhi_map:nn #1
  { \prop_put:Nnn \l__zhnum_cfg_map_ganzhi_prop { dizhi_#1 } }
\prop_new:N \l__zhnum_cfg_map_prop
\prop_new:N \l__zhnum_cfg_map_var_prop
\prop_new:N \l__zhnum_cfg_map_finan_prop
\prop_new:N \l__zhnum_cfg_map_ganzhi_prop
```

\zhnum_parse_config:
\zhnum_check_simp:nn
\zhnum_check_financial:nn
\zhnum_set_week_day:

将 prop 表转化到单独的 tl 变量。

```
\cs_new_protected:Npn \zhnum_parse_config:
```

```
  {
    \tl_clear_new:N \l__zhnum_reset_tl
    \tl_clear_new:N \l__zhnum_reset_simp_tl
    \tl_clear_new:N \l__zhnum_reset_trad_tl
    \tl_clear_new:N \l__zhnum_set_ancient_tl
    \tl_clear_new:N \l__zhnum_set_normal_tl
    \tl_clear_new:N \l__zhnum_reset_ancient_tl
    \tl_clear_new:N \l__zhnum_reset_normal_tl
    \tl_clear_new:N \l__zhnum_reset_financial_tl
    \prop_map_function:NN \l__zhnum_cfg_map_prop \zhnum_check_simp:nn
    \zhnum_set_ganzhi:
    \zhnum_reset_all:
  }
\cs_new_protected:Npn \zhnum_check_simp:nn #1#2
  {
    \prop_get:NnNTF \l__zhnum_cfg_map_var_prop
      { #1_ancient } \l__zhnum_ancient_tl
      { \__zhnum_add_reset_ancient:nN {#1} \l__zhnum_ancient_tl }
      {
        \__zhnum_check_simp_aux:nn {#2} {#1}
        \prop_get:NnNT \l__zhnum_cfg_map_finan_prop {#1} \l__zhnum_tmp_tl
          {
            \exp_args:No \__zhnum_check_simp_aux:nn
              { \l__zhnum_tmp_tl } { financial_ #1 }
            \__zhnum_add_reset_financial:n {#1}
          }
      }
  }
\cs_new_protected:Npn \__zhnum_check_simp_aux:nn #1#2
  {
    \prop_get:NnNTF \l__zhnum_cfg_map_var_prop
      { #2 _trad } \l__zhnum_trad_tl
      {
        \prop_get:NnNF \l__zhnum_cfg_map_var_prop
          { #2 _simp } \l__zhnum_simp_tl
          { \tl_set:Nn \l__zhnum_simp_tl {#1} }
        \__zhnum_add_reset_simp:nNN
          {#2} \l__zhnum_simp_tl \l__zhnum_trad_tl
      }
      { \__zhnum_add_reset:nn {#2} {#1} }
  }
\tl_new:N \l__zhnum_simp_tl
\tl_new:N \l__zhnum_trad_tl
\tl_new:N \l__zhnum_ancient_tl
\cs_new_protected:Npn \__zhnum_add_reset:nn #1#2
  {
    \tl_put_right:Nx \l__zhnum_reset_tl
      {
        \tl_set:Nn \exp_not:c { l__zhnum_ #1 _tl }
          { \exp_not:n {#2} }
      }
  }
\cs_new_protected:Npn \__zhnum_add_reset_simp:nNN #1#2#3
  {
    \tl_put_right:Nx \l__zhnum_reset_simp_tl
      {
        \tl_set:Nn \exp_not:c { l__zhnum_ #1 _tl }
          { \exp_not:o {#2} }
      }
    \tl_put_right:Nx \l__zhnum_reset_trad_tl
      {
        \tl_set:Nn \exp_not:c { l__zhnum_ #1 _tl }
          { \exp_not:o {#3} }
      }
  }
\cs_new_protected:Npn \__zhnum_add_reset_financial:n #1
  {
```

```
    \tl_put_right:Nx \l__zhnum_set_normal_tl
      {
        \tl_set_eq:NN
          \exp_not:c { l__zhnum_normal_ #1 _tl }
          \exp_not:c { l__zhnum_ #1 _tl }
      }
    \tl_put_right:Nx \l__zhnum_reset_normal_tl
      {
        \tl_set_eq:NN
          \exp_not:c { l__zhnum_ #1 _tl }
          \exp_not:c { l__zhnum_normal_ #1 _tl }
      }
    \tl_put_right:Nx \l__zhnum_reset_financial_tl
      {
        \tl_set_eq:NN
          \exp_not:c { l__zhnum_ #1 _tl }
          \exp_not:c { l__zhnum_financial_ #1 _tl }
      }
  }
\cs_new_protected:Npn \__zhnum_add_reset_ancient:nN #1#2
  {
    \tl_put_right:Nx \l__zhnum_reset_ancient_tl
      {
        \tl_set:Nn \exp_not:c { l__zhnum_ #1 _tl }
          { \exp_not:o {#2} }
      }
    \tl_put_right:Nx \l__zhnum_set_ancient_tl
      {
        \tl_concat:NNN
          \exp_not:c { l__zhnum_ #1 _tl }
          \exp_not:c { l__zhnum_   \str_head:n {#1} _tl }
          \exp_not:c { l__zhnum_ 1 \str_tail:n {#1} _tl }
      }
  }
```

\zhnum_set_week_day:
\zhnum_reset_week_day:

```
\cs_new_protected:Npn \zhnum_set_week_day:
  {
    \cs_set_protected:Npx \zhnum_reset_week_day:
      {
        \__zhnum_set_week_day:nn { mon } { 1 }
        \__zhnum_set_week_day:nn { tue } { 2 }
        \__zhnum_set_week_day:nn { wed } { 3 }
        \__zhnum_set_week_day:nn { thu } { 4 }
        \__zhnum_set_week_day:nn { fri } { 5 }
        \__zhnum_set_week_day:nn { sat } { 6 }
        \__zhnum_set_week_day:nn { sun } { day }
      }
  }
\cs_new_eq:NN \zhnum_reset_week_day: \prg_do_nothing:
\cs_new:Npn \__zhnum_set_week_day:nn #1#2
  {
    \tl_set:Nx \exp_not:c { l__zhnum_ #1 _tl }
      {
        \exp_not:N \exp_not:o { \exp_not:N \l__zhnum_weekday_tl }
        \exp_not:N \exp_not:n { \exp_not:v { l__zhnum_ #2 _tl } }
      }
  }
```

\zhnum_set_ganzhi:
\zhnum_reset_ganzhi:

```
\cs_new_protected:Npn \zhnum_set_ganzhi:
  {
    \prop_map_function:NN
      \l__zhnum_cfg_map_ganzhi_prop
      \__zhnum_add_reset:nn
  }
\cs_new_protected:Npn \__zhnum_reset_ganzhi:nn #1#2
  { \tl_set:cn { l__zhnum_ #1 _tl } {#2} }
\cs_new:Npn \zhnum_zero_mod:nn #1#2
  { \exp_args:Nf \__zhnum_zero_mod_aux:nn { \int_mod:nn {#1} {#2} } {#2} }
```

```
\cs_new:Npn \__zhnum_zero_mod_aux:nn #1#2
  { \int_compare:nNnTF {#1} = \c_zero_int {#2} {#1} }
\tl_new:c { l__zhnum_dizhi_ 0 _tl }
\tl_new:c { l__zhnum_ganzhi_ 0 _tl }
\tl_new:c { l__zhnum_tiangan_ 0 _tl }
\group_begin:
\cs_set:Npn \__zhnum_tmp:w #1
  {
    \tl_concat:NNN
      \exp_not:c { l__zhnum_ganzhi_ #1 _tl }
      \exp_not:c { l__zhnum_tiangan_ \zhnum_zero_mod:nn {#1} { 10 } _tl }
      \exp_not:c { l__zhnum_dizhi_   \zhnum_zero_mod:nn {#1} { 12 } _tl }
  }
\cs_new_protected:Npx \zhnum_reset_ganzhi:
  {
    \tl_set_eq:NN
      \exp_not:c { l__zhnum_dizhi_ 0 _tl }
      \exp_not:c { l__zhnum_dizhi_ 12 _tl }
    \tl_set_eq:NN
      \exp_not:c { l__zhnum_tiangan_ 0 _tl }
      \exp_not:c { l__zhnum_tiangan_ 10 _tl }
    \int_step_function:nN { 60 } \__zhnum_tmp:w
    \tl_set_eq:NN
      \exp_not:c { l__zhnum_ganzhi_ 0 _tl }
      \exp_not:c { l__zhnum_ganzhi_ 60 _tl }
  }
\group_end:
\cs_new_protected:Npn \zhnum_reset_config:
  { \zhnum_load_cfg:o { \l__zhnum_encoding_str } }
\cs_new_protected:Npn \zhnum_reset_all:
  {
    \zhnum_reset_main:
    \zhnum_reset_simp:
    \zhnum_set_week_day:
    \zhnum_reset_week_day:
    \zhnum_reset_ganzhi:
    \zhnum_reset_normal:
  }
\cs_new_protected:Npn \zhnum_reset_main:
  {
    \tl_use:N \l__zhnum_reset_tl
    \tl_use:N \l__zhnum_set_normal_tl
    \tl_concat:NNN
      \l__zhnum_reset_normal_tl
      \l__zhnum_reset_normal_tl
      \c__zhnum_set_zero_tl
    \tl_concat:NNN
      \l__zhnum_reset_financial_tl
      \l__zhnum_reset_financial_tl
      \c__zhnum_set_zero_tl
    \tl_concat:NNN
      \l__zhnum_reset_financial_tl
      \l__zhnum_reset_financial_tl
      \l__zhnum_set_ancient_tl
  }
\tl_const:Nx \c__zhnum_set_zero_tl
  {
    \tl_set_eq:NN
      \exp_not:N \l__zhnum_zero_tl
      \exp_not:c { l__zhnum_0_tl }
  }
\tl_new:N \l__zhnum_zero_tl
\cs_new_protected:Npn \zhnum_reset_style:
  {
    \zhnum_reset_simp:
    \zhnum_reset_normal:
  }
```

\zhnum_reset_config:
\zhnum_reset_all:
\zhnum_reset_style:

```
\cs_new_protected:Npn \zhnum_reset_simp:
  {
    \bool_if:NTF \l__zhnum_simp_bool
      { \tl_use:N \l__zhnum_reset_simp_tl }
      { \tl_use:N \l__zhnum_reset_trad_tl }
  }
\cs_new_protected:Npn \zhnum_reset_normal:
  {
    \bool_if:NTF \l__zhnum_normal_bool
      {
        \tl_use:N \l__zhnum_reset_normal_tl
        \__zhnum_reset_ancient:
        \__zhnum_reset_zero:
      }
      { \tl_use:N \l__zhnum_reset_financial_tl }
  }
\cs_new_protected:Npn \__zhnum_reset_ancient:
  {
    \bool_if:NTF \l__zhnum_ancient_bool
      { \tl_use:N \l__zhnum_reset_ancient_tl }
      { \tl_use:N \l__zhnum_set_ancient_tl }
  }
\cs_new_protected:Npx \__zhnum_reset_zero:
  {
    \exp_not:n { \bool_if:NT \l__zhnum_null_bool }
      {
        \tl_set_eq:NN
          \exp_not:c { l__zhnum_0_tl }
          \exp_not:N \l__zhnum_null_tl
      }
  }
```

\zhnum_load_cfg:n 根据选定编码载入配置文件。

```
\cs_new_protected:Npn \zhnum_load_cfg:n #1
  {
    \zhnum_set_cfg_name:Nn \l__zhnum_cfg_str {#1}
    \str_if_eq:NNTF \l__zhnum_cfg_str \l__zhnum_last_cfg_str
      { \zhnum_reset_all: }
      {
        \zhnum_update_cfg:n {#1}
        \zhnum_parse_config:
      }
  }
\cs_generate_variant:Nn \zhnum_load_cfg:n { o }
\cs_new_protected:Npn \zhnum_update_cfg:n #1
  {
    \prop_if_exist:cTF { g__zhnum_cfg_ \l__zhnum_cfg_str _prop }
      { \__zhnum_set_cfg_prop: }
      { \zhnum_input_cfg:n {#1} }
  }
\cs_new_protected:Npn \__zhnum_set_cfg_prop:
  {
    \str_set_eq:NN \l__zhnum_last_cfg_str \l__zhnum_cfg_str
    \__zhnum_update_cfg_prop:N \prop_set_eq:Nc
  }
\cs_new_protected:Npn \zhnum_input_cfg:n #1
  {
    \file_get_full_name:nNTF { zhnumber - #1 .cfg } \l__zhnum_cfg_file_tl
      {
        \bool_set_false:N \l__zhnum_reset_bool
        \__zhnum_update_cfg_prop:N \__zhnum_prop_initial:Nn
        \group_begin:
          \zhnum_set_catcode:
          \exp_args:No \file_input:n { \l__zhnum_cfg_file_tl }
          \__zhnum_update_cfg_prop:N \__zhnum_prop_gset_eq:Nn
        \group_end:
        \__zhnum_set_cfg_prop:
```

```
      }
      { \msg_error:nnx { zhnumber } { file-not-found } {#1} }
  }
\tl_new:N \l__zhnum_cfg_file_tl
\cs_new_protected:Npn \__zhnum_update_cfg_prop:N
  { \exp_args:No \__zhnum_update_cfg_prop_aux:nN { \l__zhnum_cfg_str } }
\cs_new_protected:Npn \__zhnum_update_cfg_prop_aux:nN #1#2
  {
    #2 \l__zhnum_cfg_map_prop        { g__zhnum_cfg_         #1 _prop }
    #2 \l__zhnum_cfg_map_var_prop    { g__zhnum_cfg_var_     #1 _prop }
    #2 \l__zhnum_cfg_map_finan_prop  { g__zhnum_cfg_finan_   #1 _prop }
    #2 \l__zhnum_cfg_map_ganzhi_prop { g__zhnum_cfg_ganzhi_ #1 _prop }
  }
\cs_new_protected:Npn \__zhnum_prop_initial:Nn #1#2
  {
    \prop_clear:N #1
    \prop_new:c {#2}
  }
\cs_new_protected:Npn \__zhnum_prop_gset_eq:Nn #1#2
  { \prop_gset_eq:cN {#2} #1 }
\str_new:N \l__zhnum_cfg_str
\str_new:N \l__zhnum_last_cfg_str
\bool_new:N \l__zhnum_reset_bool
\msg_new:nnnn  { zhnumber } { file-not-found }
  { File~`#1'~not~found. }
  {
    The~requested~file~could~not~be~found~in~the~current~directory,~
    in~the~TeX~search~path~or~in~the~LaTeX~search~path.
  }
```

\c__zhnum_unicode_engine_bool

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。

```
\bool_const:Nn \c__zhnum_unicode_engine_bool
  {
    \bool_lazy_any_p:n
      {
        { \sys_if_engine_xetex_p:  }
        { \sys_if_engine_luatex_p: }
        { \sys_if_engine_uptex_p: }
      }
  }
```

\zhnum_set_catcode:
\zhnum_set_cfg_name:Nn
\zhnum_reset_config:

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

```
\bool_if:NTF \c__zhnum_unicode_engine_bool
  {
    \cs_new_eq:NN \zhnum_set_catcode: \prg_do_nothing:
    \cs_new_protected:Npn \zhnum_set_cfg_name:Nn #1#2
      {
        \str_set:Nx \l__zhnum_encoding_str {#2}
        \str_set_eq:NN #1 \l__zhnum_encoding_str
      }
  }
  {
    \cs_new_protected:Npn \zhnum_set_catcode:
      {
        \bool_if:NTF \l__zhnum_active_char_bool
          { \zhnum_set_active: }
          { \zhnum_set_other: }
      }
    \cs_new_protected:Npx \zhnum_set_active:
      {
        \int_step_function:nnN
          { 128 } { 255 } \char_set_catcode_active:n
      }
    \cs_new_protected:Npx \zhnum_set_other:
      {
```

```
      \int_step_function:nnN
        { 128 } { 255 } \char_set_catcode_other:n
    }
  \cs_new_protected:Npn \zhnum_set_cfg_name:Nn #1#2
    {
      \str_set:Nx \l__zhnum_encoding_str {#2}
      \str_set:Nx #1
        {
          \l__zhnum_encoding_str
          \bool_if:NTF \l__zhnum_active_char_bool
            { / active }
            { / other }
        }
    }
  \bool_new:N \l__zhnum_active_char_bool
  \bool_set_true:N \l__zhnum_active_char_bool
}
```

\zhnum_set_encoding:n 设置编码。

```
\cs_new_protected:Npn \zhnum_set_encoding:n #1
  {
    \str_set:Nx \l__zhnum_encoding_str
      { \str_lowercase:n {#1} }
    \zhnum_load_cfg:o { \l__zhnum_encoding_str }
  }
\str_new:N \l__zhnum_encoding_str
```

encoding style null reset 宏包设置选项。

```
\keys_define:nn { zhnum / options }
  {
    encoding         .choices:nn =
      { UTF8 , GBK , Big5 }
      { \exp_args:No \zhnum_set_encoding:n { \l_keys_choice_tl } } ,
    encoding          .default:n = { GBK } ,
    encoding / Bg5       .meta:n = { encoding = Big5 } ,
    encoding / unknown   .code:n =
      { \msg_error:nnn { zhnumber } { encoding-invalid } {#1} } ,
    style .multichoice: ,
    style / Normal       .code:n =
      {
        \bool_set_false:N \l__zhnum_ancient_bool
        \bool_set_true:N  \l__zhnum_normal_bool
      } ,
    style / Financial    .code:n =
      {
        \bool_set_false:N \l__zhnum_ancient_bool
        \bool_set_false:N \l__zhnum_normal_bool
      } ,
    style / Ancient      .code:n =
      {
        \bool_set_true:N \l__zhnum_ancient_bool
        \bool_set_true:N \l__zhnum_normal_bool
      } ,
    style / Simplified   .code:n = { \bool_set_true:N  \l__zhnum_simp_bool } ,
    style / Traditional  .code:n = { \bool_set_false:N \l__zhnum_simp_bool } ,
    style             .default:n = { Normal , Simplified } ,
    style              .groups:n = { style } ,
    null              .bool_set:N = \l__zhnum_null_bool ,
    null               .groups:n = { style } ,
    time .choice: ,
    time / Chinese       .code:n = { \bool_set_true:N \l__zhnum_time_bool } ,
    time / Arabic        .code:n = { \bool_set_false:N  \l__zhnum_time_bool } ,
    time              .default:n = { Arabic } ,
    reset             .bool_set:N = \l__zhnum_reset_bool ,
    activechar        .bool_set:N = \l__zhnum_active_char_bool ,
    ganzhi-cyclic .choice: ,
```

```
    ganzhi-cyclic / true .code:n =
      { \cs_set_eq:NN \zhnum_ganzhi:nnn \zhnum_ganzhi_cyclic:nnn } ,
    ganzhi-cyclic / false.code:n =
      { \cs_set_eq:NN \zhnum_ganzhi:nnn \zhnum_ganzhi_normal:nnn } ,
    ganzhi-cyclic     .default:n = { true } ,
    arabicsep          .tl_set:N = \l__zhnum_arabic_sep_tl
  }
\bool_new:N \l__zhnum_simp_bool
\bool_new:N \l__zhnum_normal_bool
\bool_new:N \l__zhnum_ancient_bool
\msg_new:nnnn { zhnumber } { encoding-invalid }
  { The~encoding~`#1'~is~invalid. }
  { Available~encodings~are~`UTF8',~`GBK'~and~`Big5'. }
```

\zhnumsetup 在文档中设置 zhnumber 的接口。

```
\NewDocumentCommand \zhnumsetup { +m }
  {
    \zhnum_set:n {#1}
    \tex_ignorespaces:D
  }
\cs_new_protected:Npn \zhnum_set:n #1
  {
    \bool_set_false:N \l__zhnum_reset_bool
    \keys_set_filter:nnnN { zhnum / options }
      { style , user } {#1} \l__zhnum_kv_tl
    \tl_if_empty:NF \l__zhnum_kv_tl
      { \__zhnum_set_style: }
    \bool_if:NT \l__zhnum_reset_bool
      { \zhnum_reset_config: }
  }
\cs_new_protected:Npn \__zhnum_set_style:
  {
    \keys_set_filter:nnoN { zhnum / options }
      { user } { \l__zhnum_kv_tl } \l__zhnum_tmp_tl
    \tl_if_empty:NTF \l__zhnum_tmp_tl
      { \zhnum_reset_style: }
      {
        \tl_if_eq:NNF \l__zhnum_tmp_tl \l__zhnum_kv_tl
          { \zhnum_reset_style: }
        \bool_if:NF \l__zhnum_reset_bool
          { \__zhnum_set_user: }
      }
  }
\cs_new_protected:Npn \__zhnum_set_user:
  {
    \keys_set_filter:nnoN { zhnum / options }
      { pre , pos } { \l__zhnum_tmp_tl } \l__zhnum_kv_tl
    \tl_if_eq:NNF \l__zhnum_kv_tl \l__zhnum_tmp_tl
      { \zhnum_reset_normal: }
    \tl_if_empty:NF \l__zhnum_kv_tl
      { \__zhnum_set_pre_pos: }
  }
\cs_new_protected:Npn \__zhnum_set_pre_pos:
  {
    \keys_set_filter:nnoN { zhnum / options }
      { pos } { \l__zhnum_kv_tl } \l__zhnum_tmp_tl
    \tl_if_eq:NNF \l__zhnum_tmp_tl \l__zhnum_kv_tl
      {
        \zhnum_reset_week_day:
        \zhnum_reset_ganzhi:
      }
    \tl_if_empty:NF \l__zhnum_kv_tl
      { \keys_set:no { zhnum / options } { \l__zhnum_kv_tl } }
  }
\cs_generate_variant:Nn \keys_set_filter:nnnN { nno }
```

初始化设置和执行宏包选项。

```
\keys_set:nn { zhnum / options }
  { style , time , arabicsep = { ~ } }
\cs_if_exist:NTF \ProcessKeyOptions
  { \ProcessKeyOptions [ zhnum / options ] }
  {
    \RequirePackage { l3keys2e }
    \ProcessKeysOptions { zhnum / options }
  }
```

如果没有选定编码，则选用 UTF8。

```
\str_if_empty:NTF \l__zhnum_encoding_str
  { \zhnum_set_encoding:n { UTF8 } }
  { \zhnum_reset_style: }
```

⟨/package⟩

第 4 节　中文数字配置文件

⟨*config⟩

```
⟨*!big5⟩
\zhnum_set_digits_map:nnn { minus } { simp } { 负 }
\zhnum_set_digits_map:nnn { minus } { trad } { 負 }
⟨/!big5⟩
⟨*big5⟩
\zhnum_set_digits_map:nn { minus } { 負 }
⟨/big5⟩
\zhnum_set_digits_map:nn { 0 }     { 零 }
⟨*!big5⟩
\zhnum_set_digits_map:nn { null }  { 〇 }
⟨/!big5⟩
⟨*big5⟩
\zhnum_set_digits_map:nn { null }  { ○ }
⟨/big5⟩
\zhnum_set_digits_map:nn { 1 }     { 一 }
\zhnum_set_digits_map:nn { 2 }     { 二 }
\zhnum_set_digits_map:nn { 3 }     { 三 }
\zhnum_set_digits_map:nn { 4 }     { 四 }
\zhnum_set_digits_map:nn { 5 }     { 五 }
\zhnum_set_digits_map:nn { 6 }     { 六 }
\zhnum_set_digits_map:nn { 7 }     { 七 }
\zhnum_set_digits_map:nn { 8 }     { 八 }
\zhnum_set_digits_map:nn { 9 }     { 九 }
\zhnum_set_digits_map:nn { 10 }    { 十 }
\zhnum_set_digits_map:nn { 100 }   { 百 }
\zhnum_set_digits_map:nn { 1000 }  { 千 }
\zhnum_set_digits_map:nnn { 20 }   { ancient } { 廿 }
\zhnum_set_digits_map:nnn { 30 }   { ancient } { 卅 }
\zhnum_set_digits_map:nnn { 40 }   { ancient } { 卌 }
\zhnum_set_digits_map:nnn { 200 }  { ancient } { 皕 }
⟨*!big5⟩
\zhnum_set_digits_map:nnn { dot } { simp } { 点 }
\zhnum_set_digits_map:nnn { dot } { trad } { 點 }
⟨/!big5⟩
⟨*big5⟩
\zhnum_set_digits_map:nn { dot }   { 點 }
⟨/big5⟩
\zhnum_set_digits_map:nn { and }   { 又 }
\zhnum_set_digits_map:nn { parts } { 分之 }
⟨*!big5⟩
\zhnum_set_digits_map:nnn { s1 }    { simp } { 万 }
\zhnum_set_digits_map:nnn { s1 }    { trad } { 萬 }
\zhnum_set_digits_map:nnn { s2 }    { simp } { 亿 }
\zhnum_set_digits_map:nnn { s2 }    { trad } { 億 }
⟨/!big5⟩
```

```
⟨*big5⟩
\zhnum_set_digits_map:nn { s1 }    { 萬 }
\zhnum_set_digits_map:nn { s2 }    { 億 }
⟨/big5⟩
\zhnum_set_digits_map:nn { s3 }    { 兆 }
\zhnum_set_digits_map:nn { s4 }    { 京 }
\zhnum_set_digits_map:nn { s5 }    { 垓 }
\zhnum_set_digits_map:nn { s6 }    { 秭 }
\zhnum_set_digits_map:nn { s7 }    { 穰 }
⟨*!big5⟩
\zhnum_set_digits_map:nnn { s8 }    { simp } { 沟 }
\zhnum_set_digits_map:nnn { s8 }    { trad } { 溝 }
\zhnum_set_digits_map:nnn { s9 }    { simp } { 涧 }
\zhnum_set_digits_map:nnn { s9 }    { trad } { 澗 }
⟨/!big5⟩
⟨*big5⟩
\zhnum_set_digits_map:nn { s8 }    { 溝 }
\zhnum_set_digits_map:nn { s9 }    { 澗 }
⟨/big5⟩
\zhnum_set_digits_map:nn { s10 }   { 正 }
⟨*!big5⟩
\zhnum_set_digits_map:nnn { s11 }   { simp } { 载 }
\zhnum_set_digits_map:nnn { s11 }   { trad } { 載 }
⟨/!big5⟩
⟨*big5⟩
\zhnum_set_digits_map:nn { s11 }   { 載 }
⟨/big5⟩
\zhnum_set_digits_map:nn { year }  { 年 }
\zhnum_set_digits_map:nn { month } { 月 }
\zhnum_set_digits_map:nn { day }   { 日 }
⟨*!big5⟩
\zhnum_set_digits_map:nnn { hour }  { simp } { 时 }
\zhnum_set_digits_map:nnn { hour }  { trad } { 時 }
⟨/!big5⟩
⟨*big5⟩
\zhnum_set_digits_map:nn { hour }  { 時 }
⟨/big5⟩
\zhnum_set_digits_map:nn { minute }  { 分 }
\zhnum_set_digits_map:nn { weekday } { 星期 }
\zhnum_set_financial_map:nn { null } { 零 }
\zhnum_set_financial_map:nn { 0 }    { 零 }
\zhnum_set_financial_map:nn { 1 }    { 壹 }
⟨*!big5⟩
\zhnum_set_financial_map:nnn { 2 }  { simp } { 贰 }
\zhnum_set_financial_map:nnn { 2 }  { trad } { 貳 }
\zhnum_set_financial_map:nnn { 3 }  { simp } { 叁 }
\zhnum_set_financial_map:nnn { 3 }  { trad } { 叁 }
⟨/!big5⟩
⟨*big5⟩
\zhnum_set_financial_map:nn { 2 }    { 貳 }
\zhnum_set_financial_map:nn { 3 }    { 參 }
⟨/big5⟩
\zhnum_set_financial_map:nn { 4 }    { 肆 }
\zhnum_set_financial_map:nn { 5 }    { 伍 }
⟨*!big5⟩
\zhnum_set_financial_map:nnn { 6 }  { simp } { 陆 }
\zhnum_set_financial_map:nnn { 6 }  { trad } { 陸 }
⟨/!big5⟩
⟨*big5⟩
\zhnum_set_financial_map:nn { 6 }    { 陸 }
⟨/big5⟩
\zhnum_set_financial_map:nn { 7 }    { 柒 }
\zhnum_set_financial_map:nn { 8 }    { 捌 }
\zhnum_set_financial_map:nn { 9 }    { 玖 }
\zhnum_set_financial_map:nn { 10 }   { 拾 }
\zhnum_set_financial_map:nn { 100 }  { 佰 }
\zhnum_set_financial_map:nn { 1000 } { 仟 }
```

```
\zhnum_set_tiangan_map:nn { 1 }  { 甲 }
\zhnum_set_tiangan_map:nn { 2 }  { 乙 }
\zhnum_set_tiangan_map:nn { 3 }  { 丙 }
\zhnum_set_tiangan_map:nn { 4 }  { 丁 }
\zhnum_set_tiangan_map:nn { 5 }  { 戊 }
\zhnum_set_tiangan_map:nn { 6 }  { 己 }
\zhnum_set_tiangan_map:nn { 7 }  { 庚 }
\zhnum_set_tiangan_map:nn { 8 }  { 辛 }
\zhnum_set_tiangan_map:nn { 9 }  { 壬 }
\zhnum_set_tiangan_map:nn { 10 } { 癸 }
\zhnum_set_dizhi_map:nn { 1 }  { 子 }
\zhnum_set_dizhi_map:nn { 2 }  { 丑 }
\zhnum_set_dizhi_map:nn { 3 }  { 寅 }
\zhnum_set_dizhi_map:nn { 4 }  { 卯 }
\zhnum_set_dizhi_map:nn { 5 }  { 辰 }
\zhnum_set_dizhi_map:nn { 6 }  { 巳 }
\zhnum_set_dizhi_map:nn { 7 }  { 午 }
\zhnum_set_dizhi_map:nn { 8 }  { 未 }
\zhnum_set_dizhi_map:nn { 9 }  { 申 }
\zhnum_set_dizhi_map:nn { 10 } { 酉 }
\zhnum_set_dizhi_map:nn { 11 } { 戌 }
\zhnum_set_dizhi_map:nn { 12 } { 亥 }
```

⟨/config⟩

代码索引

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

K

M

N

O

P

Q

R

S

T

U

Z

zhnum internal commands: