

zhlipsum: 中文乱数假文(Lorem ipsum)

曾祥东

2018/09/08 v1.1.1*

如彭奇和瓦特曼的公共事业所证实的那样有一个胡子雪雪白的上帝超越时间超越空间确确实存在他在神圣的冷漠神圣的疯狂神圣的暗哑的高处深深地爱着我们除了少数的例外不知什么原因但时间将会揭示他像神圣的密兰达一样和人们一起忍受着痛苦这班人不知什么原因但时间将会揭示生活在痛苦中生活在烈火中这烈火这火焰如果继续燃烧毫无疑问将使穹苍着火也就是说将地狱炸上天去天是那么蓝那么澄澈那么平静这种平静尽管时断时续总比没有好得多但是别这么快还要进一步考虑到泰斯丢和丘那德的人体测定学院的未完成的研究结果早已断定毫无疑问换句话说除了依附着人类的疑问之外别无其他疑问根据泰斯丢和丘那德的未完成的劳动的结果早已作出如下的论断但是别这么快不知什么原因根据彭奇和瓦特曼的公共事业的结果已毫无疑问地断定鉴于波波夫和贝尔契不知什么原因未完成的劳动以及泰斯丢和丘那德的未完成的劳动已经就业已被许多人所否认的论点作出论断认为泰斯丢和丘那德所假设的人认为实际存在的人认为人类总而言之尽管有进步的营养学和通大便药却在衰弱萎缩衰弱萎缩而且与此同时尤其是不知什么原因尽管体育运动在各方面都有很大进展如网球足球田径车赛游泳飞行划船骑马滑翔溜冰各式各样的网球各种各样致人死命的飞行运动各式各样的秋天夏天冬天冬天网球各种各样的曲棍球盘尼西林和代用品总之我接下去讲与此同时不知什么原因要萎缩要减少尽管有网球我接下去讲飞行滑翔九穴和十八穴的高尔夫球各种各样的网球总之不知什么原因在番克汉贝克汉福尔汉克莱普汉换句话说与此同时尤其是不知什么原因但时间将会揭示要减少减少我接下去讲福尔汉克莱普汉总之自从塞缪尔·约翰逊去世以后到现在每个人的全部损失共计每人一吋四吋只是大概约略粗粗计算到小数点分量很足保持整数赤裸裸的光穿着袜子在康纳马拉总之不知什么原因不管怎样无论如何事实俱在尤其是考虑到更加远为严肃的看来更加严肃的鉴于斯丹威格和彼特曼的徒劳看来更加严肃的鉴于鉴于斯丹威格和彼特曼徒劳在平原在山地在海洋在烈火沸腾的河里天空是一样的随后是大地换句话说天空随后是大地在一片寒冷一片漆黑中天空大地石头的住所在一片寒冷中哎哟哟在我们的主诞生六百年左右天空大地海洋大地石头的住所汪洋中一片寒冷中在海上在陆地在空中我接下去讲不知什么原因尽管有网球事实俱在但时间将会揭示我接下去讲哎哟哟总之一句话石头的住所谁能怀疑我接下去讲但是别这么快我接下去讲头颅要萎缩衰弱减少与此同时尤其是不知什么原因尽管有网球胡子火焰球队石头那么蓝那么平静哎哟哟头颅头颅在康纳马拉尽管有网球未完成的徒然的劳动更加严肃的石头的住所总之我接下去讲哎哟哟徒劳的未完成的头颅头颅在康纳马拉尽管有网球头颅哎哟石头丘那德(混战,最后的狂喊)网球……石头……那么平静……丘那德……未完成的……

——萨缪尔·贝克特《等待戈多》

*<https://github.com/Stone-Zeng/zhlipsum>.

第1节 简介

`zhlipsum` 宏包用于输入中文乱数假文（拉丁语为 *Lorem ipsum*）。乱数假文是大段无意义的文字，常用来测试排版效果。支持其他语言乱数假文的宏包还有 `lipsum`、`kantlipsum`、`blindtext` 等。

`zhlipsum` 宏包支持 UTF-8、GBK 和 Big5 编码，依赖 LATEX3 项目中的 `expl3`、`xparse` 和 `l3keys2e` 宏包。为正确输入中文，`zhlipsum` 需要与 CJK 宏包或 CTEX 宏集等配套使用。

第2节 使用说明

encoding

New: 2017-09-16

Updated: 2018-04-01

`encoding = <utf8|gbk|big5>`

用于指定编码的宏包选项，可在调用宏包的时候设定。默认为 `utf8`。对于 XE_LA_TE_X、LuaL_A_TE_X 和 upL_A_TE_X 等 Unicode 引擎，`gbk` 和 `big5` 编码无效，宏包将强制使用 `utf8` 编码。

如果使用了 CTEX 宏集，则编码会根据 CTEX 自动确定。但需注意，在 CTEX 宏集中，相应的宏包选项为 `UTF8` 和 `GBK`，而本宏包中所有选项均为小写。

\zhlipsum

Updated: 2018-04-08

`\zhlipsum[<段落> [<选项>]]`
`\zhlipsum* [<段落> [<选项>]]`

插入假文文本。参数 `<段落>` 和 `<选项>` 都是可选的。注意各参数之间不可以有空格。

默认情况下，不带星号的命令 `\zhlipsum` 会在假文段落之前、之后与之间进行分段（即插入 `\par`），而带星号的命令 `\zhlipsum*` 则不做额外处理。您可以利用后文给出的 `before`、`after`、`inter` 选项来更改默认设置。

第一个可选参数 `<段落>` 为英文逗号分隔的段落编号列表，举例如下：

例 1

```
% 假设假文最大段落数为 50
\zhlipsum[2-4]          % 可用 a-b 的形式指定
\zhlipsum[4,12,3-8]    % 也可用单个数字指定
\zhlipsum[-10,40-]      % 输出 1-10 段和 40-50 段
\zhlipsum[-]            % 输出全部段落，即 1-50 段
\zhlipsum                % 默认输出 1-3 段
\zhlipsum[48-52]         % 超出部分会自动忽略，即只输出 48-50 段
```

第二个可选参数 `<选项>` 通过英文逗号分隔的键值列表形式给出。支持的选项见下。

name

New: 2018-03-24

`name = <假文名称>`

选择插入假文的名称。预定义的假文共有 6 种，见表 1。当 `encoding=utf8` 或 `gbk` 时，默认使用的假文为 `simp`；而当 `encoding=big5` 时，默认则为 `trad`。

您也可以通过 `\newzhlipsum` 命令来定义新的假文。

before

after

inter

New: 2018-03-29

`before = <内容>`
`after = <内容>`
`inter = <内容>`

在假文段落之前、之后与之间插入内容。注意使用不带星号的 `\zhlipsum` 命令时插入的分段命令会被这里的设置覆盖。

表1 预定义假文

名称	段落数	简体 / 繁体	描述	各编码下的支持情况		
				utf8	gbk	big5
simp	50	简	无意义随机假文	●	●	
trad	50	繁	无意义随机假文	●	●	●
nanshanjing	43	繁	《山海经·南山经》	●		
xiangyu	45	繁	司马迁《史记·项羽本纪》	●	●	●
zhufu	110	简	鲁迅《祝福》	●	●	
aspirin	66	简	维基百科条目:阿司匹林	●	●	

\newzhlipsum \newzhlipsum{{假文名称}}{(段落列表)}

New: 2018-03-29

声明新的假文。假文名称区分大小写。段落列表以英文逗号分隔,示例如下:

例2

```
% 注意区分中文逗号与英文逗号
\newzhlipsum{jingyesi}{%
    {床前明月光,}, {疑是地上霜。}, {举头望明月,}, {低头思故乡。}}
\zhlipsum*[-] [name=jingyesi] % 输出全部四句假文, 且不分段
```

第3节 编程接口

一般而言, 第2节中列出的命令足够一般用户使用。如需使用编程接口, 则可以考虑以下变量和函数。注意使用时需确保开启 L^AT_EX3 语法。

\g_zhlipsum_seq 假文名称列表。

\zhlipsum_use:nn 输出多段假文。

#1: 假文名称
#2: 段落编号列表

\zhlipsum_if_exist:nTF 判断是否存在对应名称的假文。

#1: 假文名称

\zhlipsum_new:nn 声明假文。

#1: 假文名称
#2: 文本列表

第4节 兼容性信息

以下选项在测试版 zhlipsum 宏包中存在,但在 1.0.0 版本之后不建议继续使用。这里仅为兼容性保留。未来将可能删除对它们的支持。

\script 过时选项。现在相当于 name。

第 5 节 已知问题

名称为 `nanshanjing` 和 `xiangyu` 的假文文本含有若干生僻字。如需正确显示, 可使用 `xeCJK` 宏包, 并设置后备字体为 `SimSun-ExtB`、`Hanazono Mincho` (花园明朝) 等, 具体方法请参考 `xeCJK` 宏包文档(仅针对编码为 UTF-8, 且使用 $\mathrm{Xe}\mathrm{\LaTeX}$ 编译的情况)。

`GBK` 和 `Big5` 编码在第二字节并没有避开 ASCII 码的范围, 因此部分汉字编码的第二字节恰好是 ASCII 编码中的一些特殊字符(如 `{`、`}`、`\` 等), 将导致编译失败。本宏包在这两种编码下的 `.def` 文件中采取了特殊技巧(见 6.6 小节), 请避免修改这些文件。

如无特殊需要, 始终建议您采用 `UTF-8` 编码, 并使用 $\mathrm{Xe}\mathrm{\LaTeX}$ 、 $\mathrm{Lua}\mathrm{\LaTeX}$ 等 `Unicode` 引擎编译。

特殊情况下, 如果您必须使用 `GBK` 或 `Big5` 编码, 并需要声明新的假文, 可以采取以下手段临时回避上述问题。

例 3

```
% 文件编码需使用 Big5
% \usepackage[encoding=big5]{zhlipsum}

% 直接使用 \newzhlipsum{big5}{許蓋功, 蓋功許, 功許蓋} 会报错
% 原理: 在分组内用 <、>、+ 代替 {、}、\, 再将原先的 {、}、\ 设为“其他”
% 类 (即类别码为 12)
\begingroup
  \catcode`\<=1
  \catcode`\>=2
  \catcode`\+=0
  \catcode`\{=12
  \catcode`\}=12
  \catcode`\\\=12
  +newzhlipsum<big5><許蓋功, 蓋功許, 功許蓋>
+endgroup
\zhlipsum[name=big5]
```

第 6 节 实现细节

```
1 <*package>
2 <@=@=zhlipsum>

    检查  $\mathrm{L}\mathrm{\TeX}3$  编程环境。
3 \RequirePackage { xparse, l3keys2e }
4 \msg_new:nnn { zhlipsum } { 13-too-old }
5  {
6     Package~ "#1"~ is~ too~ old. \\ \\
7     Please~ update~ an~ up-to-date~ version~ of~ the~ bundles \\
8     "l3kernel"~ and~ "l3packages"~ using~ your~ TeX~ package \\
9     manager~ or~ from~ CTAN.
10 }
11 \clist_map_inline:nn { expl3, xparse, l3keys2e }
12  {
13     \ifpackagelater {#1} { 2018/05/12 }
14     { } { \msg_error:nnn { zhlipsum } { 13-too-old } {#1} }
15 }
```

6.1 内部变量和函数

临时变量。

```
16 \tl_new:N \l_zhlipsum_tmpa_tl
17 \seq_new:N \l_zhlipsum_tmpa_seq
```

18 \seq_new:N \l_zhlipsum_tmpb_seq

\g_zhlipsum_encoding_tl 编码信息。

19 \tl_new:N \g_zhlipsum_encoding_tl

\g_zhlipsum_seq 假文名称列表。

20 \seq_new:N \g_zhlipsum_seq

\c_zhlipsum_simp_seq 预定义的简体中文与繁体中文的假文名称列表。

\c_zhlipsum_trad_seq
 21 \seq_new:N \c_zhlipsum_simp_seq
 22 \seq_new:N \c_zhlipsum_trad_seq
 23 \seq_set_from_clist:Nn \c_zhlipsum_simp_seq { simp, zhufu, aspirin }
 24 \seq_set_from_clist:Nn \c_zhlipsum_trad_seq { trad, xiangyu, nanshanjing }

LATEX3 函数变体。

25 \cs_generate_variant:Nn \file_input:n { x }

26 \prg_generate_conditional_variant:Nnn \tl_if_eq:nn { Vn } { T, TF }

_zhlipsum_if_unicode_engine:TF 判断是否为 Unicode 引擎。来自于 zhnumber 宏包。

27 \prg_new_protected_conditional:Npnn _zhlipsum_if_unicode_engine: { T, F, TF }
 28 {
 29 \bool_lazy_any:nTF
 30 {
 31 { \sys_if_engine_xetex_p: }
 32 { \sys_if_engine_luatex_p: }
 33 { \sys_if_engine_uptex_p: }
 34 }
 35 { \prg_return_true: }
 36 { \prg_return_false: }
 37 }

_zhlipsum_if_encoding:nTF 判断当前编码。

38 \prg_new_protected_conditional:Npnn _zhlipsum_if_encoding:n #1 { T, F, TF }
 39 {
 40 \tl_if_eq:VnTF \g_zhlipsum_encoding_tl {#1}
 41 { \prg_return_true: } { \prg_return_false: }
 42 }

_zhlipsum_msg_new:nn 各种信息函数的缩略形式。

43 \cs_new:Npn _zhlipsum_msg_new:nn { \msg_new:nnn { zhlipsum } }
 44 \cs_new:Npn _zhlipsum_error:n { \msg_error:nn { zhlipsum } }
 45 \cs_new:Npn _zhlipsum_error:nn { \msg_error:nnn { zhlipsum } }
 46 \cs_new:Npn _zhlipsum_warning:nn { \msg_warning:nnn { zhlipsum } }
 47 \cs_new:Npn _zhlipsum_warning:nnnn { \msg_warning:nnnn { zhlipsum } }
 48 \cs_new:Npn _zhlipsum_warning:nxxx { \msg_warning:nnxxx { zhlipsum } }
 49 \cs_new:Npn _zhlipsum_info:nn { \msg_info:nnn { zhlipsum } }

_zhlipsum_par: 分段命令。

50 \cs_new_eq:NN _zhlipsum_par: \tex_par:D

6.2 宏包选项

`encoding` 设置编码。

```

51 \keys_define:nn { zhlipsum / option }
52   {
53     encoding .choices:nn =
54     { utf8, gbk, big5 }
55   {
56     \tl_gset_eq:NN \g__zhlipsum_encoding_tl \l_keys_choice_tl
57     \__zhlipsum_if_unicode_engine:T
58     {
59       \__zhlipsum_if_encoding:nF { utf8 }
60       {
61         \tl_gset:Nn \g__zhlipsum_encoding_tl { utf8 }
62         \__zhlipsum_warning:nn { unicode-engine } {#1}
63       }
64     }
65     \__zhlipsum_if_ctex_valid_encoding:F
66     {
67       \__zhlipsum_error:nn { ctex-invalid-encoding } {#1} }
68   },
69   encoding / unknown .code:n =
70   {
71     \__zhlipsum_error:nn { invalid-encoding } {#1} },
72   encoding .value_required:n = true,
```

处理未知选项。

```

71   unknown .code:n = { \__zhlipsum_error:n { unknown-option } }
72 }
```

提示信息。

```

73 \__zhlipsum_msg_new:nn { unicode-engine }
74 {
75   You~ are~ now~ using~ Unicode~ engine~ \c_sys_engine_str. \\
76   Encoding~ "#1"~ is~ invalid.~ Changed~ into~ "utf8".
77 }
78 \__zhlipsum_msg_new:nn { ctex-invalid-encoding }
79 {
80   Package~ option~ "encoding=#1"~ is~ in~ conflict~ with~ ctex's~
81   option~ "\tl_use:N \l__ctex_encoding_tl".\\\\
82   Please~ check~ the~ package~ options.
83 }
84 \__zhlipsum_msg_new:nn { invalid-encoding }
85 {
86   Encoding~ "#1"~ is~ invalid. \\
87   Available~ encodings~ are~ "utf8",~ "gbk"~ and~ "big5".
88 }
89 \__zhlipsum_msg_new:nn { unknown-option }
90 { Package~ option~ "\l_keys_key_tl"~ is~ unknown. }
```

`__zhlipsum_if_ctex_valid_encoding:TF` 检查 CTeX 编码。

```

91 \prg_new_protected_conditional:Npn \__zhlipsum_if_ctex_valid_encoding: { F }
92 {
93   \tl_if_exist:NTF \l__ctex_encoding_tl
94   {
95     \tl_set:Nx \l__zhlipsum_tmpa_tl
96     { \str_lower_case:f { \l__ctex_encoding_tl } }
97     \str_if_eq:NNTF \g__zhlipsum_encoding_tl \l__zhlipsum_tmpa_tl
98     { \prg_return_true: } { \prg_return_false: }
99   }
100 { \prg_return_true: }
```

如果调用了 CTEX 宏集，则自动确定编码；否则默认设为 UTF-8。

```

102 \tl_if_exist:NNTF \l__ctex_encoding_tl
103 {
104     \tl_if_eq:VnTF \l__ctex_encoding_tl { UTF8 }
105     { \tl_gset:Nn \g__zhlipsum_encoding_tl { utf8 } }
106     {
107         \tl_if_eq:VnT \l__ctex_encoding_tl { GBK }
108         { \tl_gset:Nn \g__zhlipsum_encoding_tl { gbk } }
109     }
110 }
111 { \tl_gset:Nn \g__zhlipsum_encoding_tl { utf8 } }

将宏包选项传给 zhlipsum/option。
112 \ProcessKeysOptions { zhlipsum / option }
```

6.3 函数选项

\l__zhlipsum_name_tl 保存假文名称。

```
113 \tl_new:N \l__zhlipsum_name_tl
```

\l__zhlipsum_before_tl 保存假文之前、之后与之间插入的内容。

```

114 \tl_new:N \l__zhlipsum_before_tl
115 \tl_new:N \l__zhlipsum_after_tl
116 \tl_new:N \l__zhlipsum_inter_tl

117 \keys_define:nn { zhlipsum }
118 {
```

name 假文名称。Big5 编码不支持简体中文。

```

119 name .code:n =
120 {
121     \tl_set:Nn \l__zhlipsum_name_tl {#1}
122     \__zhlipsum_if_encoding:nT { big5 }
123     {
124         \seq_if_in:NVT \c_zhlipsum_simp_seq \l__zhlipsum_name_tl
125         {
126             \__zhlipsum_warning:nn { big5-require-trad } {#1}
127             \tl_set:Nn \l__zhlipsum_name_tl { trad }
128         }
129     }
130 },
```

script 选择输入简体中文或繁体中文。过时选项。

```

131 script .code:n =
132 {
133     \__zhlipsum_warning:nn { deprecated-option }
134     { Option~ "name=#1"~ will~ be~ set. }
135     \keys_set:nn { zhlipsum } { name = #1 }
136 },
```

before 假文之前、之后与之间插入的内容。

```

137 before .tl_set:N = \l__zhlipsum_before_tl,
138 after .tl_set:N = \l__zhlipsum_after_tl,
139 inter .tl_set:N = \l__zhlipsum_inter_tl
140 }
```

提示信息。

```

141 \__zhlipsum_msg_new:nn { big5-require-trad }
142 {
143     Name~ "#1"~ is~ not~ available~ in~ "Big5"~ encoding. \\
144     Changed~ into~ "trad".
145 }
146 \__zhlipsum_msg_new:nn { deprecated-option }
147 { Option~ "\l_keys_key_tl"~ is~ deprecated. \\ #1 }

初始选项设置。
148 \__zhlipsum_if_encoding:nTF { big5 }
149 { \keys_set:nn { zhlipsum } { name = trad } }
150 { \keys_set:nn { zhlipsum } { name = simp } }

```

6.4 输出假文

`\zhlipsum` 输出假文,第一个可选参数表示段落数,默认为 1-3;第二个可选参数为选项列表。

```

151 \NewDocumentCommand \zhlipsum { s o +o }
152 {
153     \group_begin:
154     \IfBooleanF {#1}
155     {
156         \tl_set:Nn \l__zhlipsum_before_tl { \__zhlipsum_par: }
157         \tl_set:Nn \l__zhlipsum_after_tl { \__zhlipsum_par: }
158         \tl_set:Nn \l__zhlipsum_inter_tl { \__zhlipsum_par: }
159     }
160     \IfValueTF {#3}
161     {
162         \keys_set:nn { zhlipsum } {#3}
163         \zhlipsum_use:Vn \l__zhlipsum_name_tl {#2}
164     }
165     {
166         \IfValueTF {#2}
167         {

```

如果只带一个参数,那么根据其是否含有 = 来判断该参数是段落数还是选项列表。

```

168     \__zhlipsum_if_key_value_list:nTF {#2}
169     {
170         \keys_set:nn { zhlipsum } {#2}
171         \zhlipsum_use:Vn \l__zhlipsum_name_tl { 1 - 3 }
172     }
173     { \zhlipsum_use:Vn \l__zhlipsum_name_tl {#2} }
174 }
175 { \zhlipsum_use:Vn \l__zhlipsum_name_tl { 1 - 3 } }
176 }
177 \group_end:
178 }
```

`__zhlipsum_if_key_value_list:nTF` 判断是否为键值列表,即是否含有 =。

```

179 \prg_new_protected_conditional:Npnn \__zhlipsum_if_key_value_list:n #1 { TF }
180 {
181     \tl_if_in:nnTF {#1} { = }
182     { \prg_return_true: } { \prg_return_false: }
183 }
```

`\l__zhlipsum_par_num_seq` 保存段落编号。

```
184 \seq_new:N \l__zhlipsum_par_num_seq
```

\zhlipsum_use:nn
\zhlipsum_use:Vn
输出多段假文。#1 = 假文名称, #2 = 段落编号列表。解析段落编号之后, 按次序逐项输出, 并在前后插入相应内容。注意最后一段需要单独处理。

```

185 \cs_new_protected:Npn \zhlipsum_use:nn #1#2
186 {
187     \__zhlipsum_if_cjk_valid_encoding:TF
188     {
189         \zhlipsum_if_exist:nTF {#1}
190         {
191             \__zhlipsum_parse_par:nn {#1} {#2}
192             \seq_pop_right:NN \l__zhlipsum_par_num_seq \l__zhlipsum_tmpa_tl
193             \tl_use:N \l__zhlipsum_before_tl
194             \seq_map_inline:Nn \l__zhlipsum_par_num_seq
195             {
196                 \__zhlipsum_use:nn {#1} {##1}
197                 \tl_use:N \l__zhlipsum_inter_tl
198             }
199             \__zhlipsum_use:nn {#1} { \tl_use:N \l__zhlipsum_tmpa_tl }
200             \tl_use:N \l__zhlipsum_after_tl
201         }
202         { \__zhlipsum_error:nn { invalid-name } {#1} }
203     }
204     { \__zhlipsum_error:n { CJK-invalid-encoding } }
205 }
206 \cs_generate_variant:Nn \zhlipsum_use:nn { Vn }
207 \__zhlipsum_msg_new:nn { invalid-name }
208 {
209     Name~ "#1"~ is~ unknown. \\\\
210     Please~ use~ the~ pre-defined~ Chinese~ dummy~ texts~ or~
211     declare~ new~ one.
212 }
213 \__zhlipsum_msg_new:nn { CJK-invalid-encoding }
214 {
215     The~ current~ CJK~ environment~ uses~ "\tl_use:N \CJK@@@enc"~
216     encoding, \\
217     but~ zhlipsum~ package~ has~ been~ loaded~ with~ the~ option~
218     "encoding=\tl_use:N \g__zhlipsum_encoding_tl".\\\
219     Please~ check~ the~ package~ options.
220 }
```

__zhlipsum_if_cjk_valid_encoding:TF 检查 CJK 环境编码。

```

221 \prg_new_protected_conditional:Npnn \__zhlipsum_if_cjk_valid_encoding: { TF }
222 {
223     \tl_if_exist:NTF \CJK@@@enc
224     {
225         \tl_set:Nx \l__zhlipsum_tmpa_tl { \str_lower_case:f { \CJK@@@enc } }
226 % TODO: str or tl?
227         \str_if_eq:NNTF \g__zhlipsum_encoding_tl \l__zhlipsum_tmpa_tl
228             { \prg_return_true: }
229             {
230                 \__zhlipsum_if_encoding:nTF { gbk }
231                 {
232                     \str_if_eq:VnTF \l__zhlipsum_tmpa_tl { gb }
233                         { \prg_return_true: } { \prg_return_false: }
234                 }
235                 {
236                     \__zhlipsum_if_encoding:nTF { big5 }
237                     {
238                         \str_if_eq:VnTF \l__zhlipsum_tmpa_tl { bg5 }
239                             { \prg_return_true: } { \prg_return_false: }
240                         }
241                     { \prg_return_false: }
242                 }
```

```

243         }
244     }
245     { \prg_return_true: }
246 }

```

\zhlipsum_if_exist:nTF

```

247 \prg_new_protected_conditional:Npnn \zhlipsum_if_exist:n #1 { T, F, TF }
248 {
249     \seq_if_in:NnTF \g_zhlipsum_seq {#1}
250     { \prg_return_true: } { \prg_return_false: }
251 }

```

```

\l_zhlipsum_begin_int 252 \int_new:N \l_zhlipsum_begin_int
\l_zhlipsum_end_int 253 \int_new:N \l_zhlipsum_end_int
\l_zhlipsum_max_int 254 \int_new:N \l_zhlipsum_max_int

```

```

\l_zhlipsum_modified_range_bool 255 \bool_new:N \l_zhlipsum_modified_range_bool
\l_zhlipsum_invalid_range_bool 256 \bool_new:N \l_zhlipsum_invalid_range_bool

```

_zhlipsum_parse_par:nn

解析段落编号列表。#1 = 假文名称, #2 = 段落编号列表。
编号列表用逗号分隔, 其中的每一项为单个数字或为 a-b 的形式。若 a、b 为空, 则分别取为 1 或允许的最大值(即段落数)。超过范围的数字则忽略。

```

257 \cs_new_protected:Npn \_zhlipsum_parse_par:nn #1#2
258 {
259     \seq_clear:N \l_zhlipsum_par_num_seq
260     \int_set_eq:Nc \l_zhlipsum_max_int { g_zhlipsum_ #1 _int }
261     \clist_map_inline:nn {#2}
262     {
263         \_zhlipsum_parse_par_aux:n {##1}
264         \bool_if:NTF \l_zhlipsum_invalid_range_bool
265             { \_zhlipsum_warning:nnn { invalid-range } {##1} {#2} }
266             {
267                 \bool_if:NT \l_zhlipsum_modified_range_bool
268                     {
269                         \_zhlipsum_warning:nxxx { modified-range }
270                         {##1} {#2} { \_zhlipsum_par_range: }
271                     }
272                     \seq_concat:NNN \l_zhlipsum_par_num_seq
273                         \l_zhlipsum_par_num_seq \l_zhlipsum_tmpa_seq
274                 }
275             }
276     }

```

```

\_\zhlipsum_parse_par_aux:n 277 \cs_new_protected:Npn \_\zhlipsum_parse_par_aux:n #1
278 {
279     \bool_set_false:N \l_zhlipsum_modified_range_bool
280     \bool_set_false:N \l_zhlipsum_invalid_range_bool
281     \seq_clear:N \l_zhlipsum_tmpa_seq
282     \tl_if_in:nnTF {#1} { - }
283     {
284         \seq_set_split:Nnn \l_zhlipsum_tmpb_seq { - } {#1}

```

“-”左侧的数字。

```

285     \seq_pop_left:NN \l_zhlipsum_tmpb_seq \l_zhlipsum_tmpa_tl
286     \tl_if_empty:NTF \l_zhlipsum_tmpa_tl
287         { \int_set_eq:NN \l_zhlipsum_begin_int \c_one_int }
288         {
289             \int_set:Nn \l_zhlipsum_begin_int { \l_zhlipsum_tmpa_tl }
290             \int_compare:nNnT \l_zhlipsum_begin_int < \c_one_int
291                 {
292                     \int_set_eq:NN \l_zhlipsum_begin_int \c_one_int
293                     \bool_set_true:N \l_zhlipsum_modified_range_bool

```

```

294     }
295 }
```

“-”右侧的数字。注意左右数字均由 \seq_pop_left:NN 得到，因此 -3-4 实际相当于 -3，进而被解析为 1-3。

```

296     \seq_pop_left:NN \l_zhlipsum_tmpb_seq \l_zhlipsum_tma_t1
297     \tl_if_empty:NTF \l_zhlipsum_tma_t1
298     { \int_set_eq:NN \l_zhlipsum_end_int \l_zhlipsum_max_int }
299     {
300         \int_set:Nn \l_zhlipsum_end_int { \l_zhlipsum_tma_t1 }
301         \int_compare:nNnT \l_zhlipsum_end_int > \l_zhlipsum_max_int
302         {
303             \int_set_eq:NN \l_zhlipsum_end_int \l_zhlipsum_max_int
304             \bool_set_true:N \l_zhlipsum_modified_range_bool
305         }
306     }
```

检查取值范围。

```

307     \bool_lazy_or:nnTF
308     { \int_compare_p:nNn \l_zhlipsum_begin_int > \l_zhlipsum_max_int }
309     { \int_compare_p:nNn \l_zhlipsum_begin_int > \l_zhlipsum_end_int }
310     { \bool_set_true:N \l_zhlipsum_invalid_range_bool }
311     {
312         \int_step_inline:nnn
313         { \l_zhlipsum_begin_int } { \l_zhlipsum_end_int }
314         { \seq_put_right:Nn \l_zhlipsum_tma_seq {##1} }
315     }
316 }
```

单个数字的处理。

```

318     \bool_lazy_or:nnTF
319     { \int_compare_p:nNn {#1} > \l_zhlipsum_max_int }
320     { \int_compare_p:nNn {#1} < \c_one_int }
321     { \bool_set_true:N \l_zhlipsum_invalid_range_bool }
322     { \seq_put_right:Nn \l_zhlipsum_tma_seq {#1} }
323 }
```

__zhlipsum_par_range: 显示段落范围(用在提示信息中,可以完全展开)。

```

325 \cs_new:Npn \__zhlipsum_par_range:
326 {
327     \int_compare:nNnTF \l_zhlipsum_begin_int = \l_zhlipsum_end_int
328     { \int_use:N \l_zhlipsum_begin_int }
329     { \int_use:N \l_zhlipsum_begin_int - \int_use:N \l_zhlipsum_end_int }
330 }
```

提示信息。

```

331 \__zhlipsum_msg_new:nn { modified-range }
332 {
333     Your~ required~ range~ "#1"~ in~ "#2"~ will~ be~ modified. \\
334     Changed~ into~ "#3".
335 }
336 \__zhlipsum_msg_new:nn { invalid-range }
337 {
338     Your~ required~ range~ "#1"~ in~ "#2"~ is~ invalid. \\
339     Nothing~ will~ be~ output.
340 }
```

__zhlipsum_use:nn 输出一段假文。#1 = 假文名称, #2 = 段落编号。

```

341 \cs_new_protected:Npn \__zhlipsum_use:nn #1#2
342     { \tl_use:c { c_zhlipsum_ #1 @ #2 _tl } }
```

6.5 声明假文

声明假文。#1 = 假文名称, #2 = 文本 clist。

```

343 \NewDocumentCommand \newzhlipsum { m m }
344   { \zhlipsum_new:nn {#1} {#2} }
345 \cs_new_protected:Npn \zhlipsum_new:nn #1#2
346   {
347     \zhlipsum_if_exist:nTF {#1}
348       { \__zhlipsum_error:nn { already-defined } {#1} }
349       {
350         \seq_gput_left:Nn \g_zhlipsum_seq {#1}
351         \int_new:c { g_zhlipsum_ #1 _int }
352         \clist_map_inline:nn {#2} { \__zhlipsum_new:nn {#1} {##1} }
353         \__zhlipsum_info:nn { defining-text } {#1}
354       }
355     }
356 \__zhlipsum_msg_new:nn { already-defined }
357   {
358     Chinese~ dummy~ text~ "#1"~ has~ been~ used.~
359     Please~ use~ another~ name.
360   }
361 \__zhlipsum_msg_new:nn { defining-text }
362   {
363     Chinese~ dummy~ text~ "#1"~ is~ created.~
364     It~ has~ \int_use:c { g_zhlipsum_ #1 _int }~ paragraphs.
365   }

```

__zhlipsum_new:nn 定义新的假文段落。#1 = 假文名称, #2 = 文本。

```

366 \cs_new_protected:Npn \__zhlipsum_new:nn #1#2
367   {
368     \int_gincr:c { g_zhlipsum_ #1 _int }
369     \tl_const:cn
370     { c_zhlipsum_ #1 @ \int_use:c { g_zhlipsum_ #1 _int } _tl } {#2}
371   }

```

根据编码读入假文文本定义文件。

```

372 \file_input:x { zhlipsum- \g_zhlipsum_encoding_tl .def }
373 </package>

```

6.6 假文文本

__zhlipsum_set_special_catcode: 在声明预定义文本时,为了兼容 CJK 宏包的特殊处理,需要临时更改类别码。具体来说,在 GBK/Big5 编码下,由于汉字的第二个字节会与 TeX 中的特殊符号 \、{、}、~ 冲突,所以需要将它们的类别码改为 12(其他),并分别用 +、<、> 和 * 代替。星号 * 在开启 LATEX3 语法后实际相当于空格(类别码为 10)。

```

374 <*text>
375 \cs_new_protected:Npn \__zhlipsum_set_special_catcode:
376   {
377     \__zhlipsum_active_first_byte:
378     \char_set_catcode_escape:N      +
379     \char_set_catcode_group_begin:N <
380     \char_set_catcode_group_end:N   >
381     \char_set_catcode_space:N      *
382     \char_set_catcode_other:N    \\
383     \char_set_catcode_other:N  \{
384     \char_set_catcode_other:N  \}
385     \char_set_catcode_other:N  ~
386   }

```

__zhlipsum_active_first_byte: 将汉字的首字节设为活动字符(类别码 12)。UTF-8 编码下不需要该操作。

```
387 <!*utf8>
388 \cs_new_protected:Npx \_\_zhlipsum\_active\_first\_byte:
389   {
390     \int_step_function:nnN { "81" } { "FE" }
391     \char_set_catcode_active:n
392   }
393 </!utf8>
394 </text>
```

预定义假文的声明放置在分组内，利用 __zhlipsum_set_special_catcode: 切换类别码后可以不再需要 CJK 的预处理操作。具体声明此处不再列出。

版本历史

v0.1	(2017/04/08)	v1.0.0	(2018/03/23 – 2018/04/01)				
General: 开始编写宏包。	1	General: script 成为过时选项。	7				
v0.2	(2017/04/14)	根据 CTeX 宏集的选项自动确定编码。	6				
General: 仿照 kantlipsum 宏包, 实现任意的段落选取。	1	利用类别码机制回避 CJK 宏包的预处理操作。	12				
使用名字空间。	1	新增选项 before、after。	7				
v0.4	(2017/09/16)	新增选项 inter。	7				
General: 将安装、测试文件集成进源文件。	1	新增选项 name。	7				
新增 encoding 选项。	6	增加预定义假文。	13				
优化宏包实现。	1	\zhlipsum: 更改参数形式, 允许利用逗号分隔列表选择段落。	8				
v0.5	(2017/12/22 – 2018/01/06)	v1.1.0	(2018/04/08)				
General: 添加英文版用户文档。	1	\zhlipsum: 改回使用方括号指定段落数的形式。	8				
新增选项 script, 同时支持简体中文和繁体中文。	7	v1.1.1	(2018/07/19 – 2018/09/08)				
支持 Big5 编码。	6	\zhlipsum: 更新假文文本。	13	\zhlipsum: 支持选项设置。	8	完善持续集成系统, 在多平台上进行测试。	1
\zhlipsum: 更新假文文本。	13						
\zhlipsum: 支持选项设置。	8	完善持续集成系统, 在多平台上进行测试。	1				

代码索引

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

Symbols	
*	381
\+	378
\<	379
\{	383
\}	384
\~	385
A	
after	2, <u>137</u>
B	
before	2, <u>137</u>
bool commands:	
\bool_if:NTF	264, 267
\bool_lazy_any:nTF	29
\bool_lazy_or:nnTF	307, 318
\bool_new:N	255, 256
\bool_set_false:N	279, 280
\bool_set_true:N	293, 304, 310, 321
C	
char commands:	
\char_set_catcode_active:n	391
\char_set_catcode_escape:N	378
\char_set_catcode_group_begin:N	379
\char_set_catcode_group_end:N	380
E	
encoding	2, <u>51</u>
F	
file commands:	
\file_input:n	25, 372
G	
group commands:	
\group_begin:	153
\group_end:	177
I	
\IfBooleanF	154

\IfValueTF	160, 166	\seq_concat:NNN	272
int commands:		\seq_gput_left:Nn	350
\int_compare:nNnTF	290, 301, 327	\seq_if_in:NnTF	124, 249
\int_compare_p:nNn	308, 309, 319, 320	\seq_map_inline:Nn	194
\int_gincr:N	368	\seq_new:N	17, 18, 20, 21, 22, 184
\int_new:N	252, 253, 254, 351	\seq_pop_left>NN	11, 285, 296
\int_set:Nn	289, 300	\seq_pop_right:NN	192
\int_set_eq:NN	260, 287, 292, 298, 303	\seq_put_right:Nn	314, 322
\int_step_function:nnN	390	\seq_set_from_clist:Nn	23, 24
\int_step_inline:nnn	312	\seq_set_split:Nnn	284
\int_use:N	328, 329, 364, 370	\g_zhlipsum_seq	3, 20, 249, 350
\c_one_int	287, 290, 292, 320		
inter	2, 137		
K			
keys commands:			
\l_keys_choice_tl	56	\str_if_eq:NNTF	97, 227
\keys_define:nn	51, 117	\str_if_eq:nnTF	232, 238
\l_keys_key_tl	90, 147	\str_lower_case:n	96, 225
\keys_set:nn	135, 149, 150, 162, 170		
M			
msg commands:			
\msg_error:nn	44	\c_sys_engine_str	75
\msg_error:nnn	14, 45	\sys_if_engine_luatex_p:	32
\msg_info:nnn	49	\sys_if_engine_uptex_p:	33
\msg_new:nnn	4, 43	\sys_if_engine_xetex_p:	31
\msg_warning:nnn	46		
\msg_warning:nnnn	47		
\msg_warning:nnnnn	48		
N			
name	2, 119		
\NewDocumentCommand	151, 343		
\newzhlipsum	2, 3, 343		
P			
prg commands:			
\prg_generate_conditional_variant:Nnn	26	\tl_const:Nn	369
\prg_new_protected_conditional:Npnn	27, 38, 91, 179, 221, 247	\tl_gset:Nn	61, 105, 108, 111
\prg_return_false:	36, 41, 98, 182, 233, 239, 241, 250	\tl_gset_eq:NN	56
\prg_return_true:	35, 41, 98, 100, 182, 228, 233, 239, 245, 250	\tl_if_empty:NTF	286, 297
\ProcessKeysOptions	112	\tl_if_eq:nn	26
R			
\RequirePackage	3	\tl_if_eq:nnTF	40, 104, 107
S			
script	3, 131	\tl_if_exist:NTF	93, 102, 223
seq commands:		\tl_if_in:nnTF	181, 282
\seq_clear:N	259, 281	\tl_new:N	16, 19, 113, 114, 115, 116
		\tl_set:Nn	95, 121, 127, 156, 157, 158, 225
		\tl_use:N	81, 193, 197, 199, 200, 215, 218, 342
Z			
\zhlipsum	2, 2, 151		
zhlipsum commands:			
\zhlipsum_if_exist:nTF	3, 189, 247, 347		
\zhlipsum_new:nn	3, 343		
\c_zhlipsum_simp_seq	21, 124		
\c_zhlipsum_trad_seq	21		
\zhlipsum_use:nn	3, 163, 171, 173, 175, 185		
zhlipsum internal commands:			
__zhlipsum_active_first_byte:	377, 387		
\l_zhlipsum_after_tl	114, 138, 157, 200		
\l_zhlipsum_before_tl	114, 137, 156, 193		

```
\l__zhlipsum_begin_int ..... 252,  
    287, 289, 290, 292, 308, 309, 313, 327, 328, 329  
\g__zhlipsum_encoding_tl .....  
    ... 19, 40, 56, 61, 97, 105, 108, 111, 218, 227, 372  
\l__zhlipsum_end_int .....  
    ... 252, 298, 300, 301, 303, 309, 313, 327, 329  
\__zhlipsum_error:n ..... 43, 71, 204  
\__zhlipsum_error:nn ..... 43, 66, 69, 202, 348  
\__zhlipsum_if_cjk_valid_encoding:TF 187, 221  
\__zhlipsum_if_ctex_valid_encoding:TF 65, 91  
\__zhlipsum_if_encoding:nTF .....  
    ... 38, 59, 122, 148, 230, 236  
\__zhlipsum_if_key_value_list:nTF . 168, 179  
\__zhlipsum_if_unicode_engine:TF ... 27, 57  
\__zhlipsum_info:nn ..... 43, 353  
\l__zhlipsum_inter_tl ..... 114, 139, 158, 197  
\l__zhlipsum_invalid_range_bool .....  
    ... 255, 264, 280, 310, 321  
\l__zhlipsum_max_int .....  
    ... 252, 260, 298, 301, 303, 308, 319  
\l__zhlipsum_modified_range_bool .....
```



```
..... 255, 267, 279, 293, 304  
\__zhlipsum_msg_new:nn ..... 43,  
    73, 78, 84, 89, 141, 146, 207, 213, 331, 336, 356, 361  
\l__zhlipsum_name_tl .....  
    ... 113, 121, 124, 127, 163, 171, 173, 175  
\__zhlipsum_new:nn ..... 352, 366  
\__zhlipsum_par: ..... 50, 156, 157, 158  
\l__zhlipsum_par_num_seq .....  
    ... 184, 192, 194, 259, 272, 273  
\__zhlipsum_par_range: ..... 270, 325  
\__zhlipsum_parse_par:nn ..... 191, 257  
\__zhlipsum_parse_par_aux:n ..... 263, 277  
\__zhlipsum_set_special_catcode: ... 13, 374  
\l__zhlipsum_tmpa_seq ... 16, 273, 281, 314, 322  
\l__zhlipsum_tmpa_tl ..... 16, 95, 97, 192,  
    199, 225, 227, 232, 238, 285, 286, 289, 296, 297, 300  
\l__zhlipsum_tmpb_seq ..... 16, 284, 285, 296  
\__zhlipsum_use:nn ..... 196, 199, 341  
\__zhlipsum_warning:nn ..... 43, 62, 126, 133  
\__zhlipsum_warning:nnn ..... 43, 265  
\__zhlipsum_warning:nnnn ..... 43, 269
```