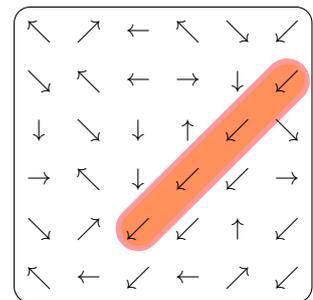
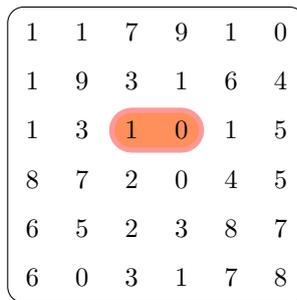
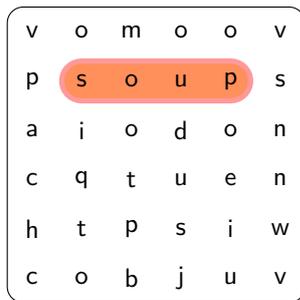


# The `soup` package <sup>\*</sup>

Thomas Simers <sup>†</sup>

Released 2019/04/05



## Abstract

The goal of `soup` is to generate the grid of letters for a word search, puzzle sometimes called “alphabet soup” (from which this package gets its name) or “find-the-word.”

In addition to supporting classic word searches, the `soup` can be filled with numbers or a user-defined set of glyphs.

Full functionality relies on TikZ, but limited support without TikZ is available through a package option.

---

<sup>\*</sup>This document corresponds to `soup` v1.0.2, last revised 2019/04/05.

<sup>†</sup>E-mail: [Simers.T@gmail.com](mailto:Simers.T@gmail.com)

# 1 User Guide

The soup interface rests primarily in two parts: The environments which determine the type of soup (alphabet, number, or homemade), and the shared macros for inserting and marking clues.

## 1.1 Load-Time Options

---

`usetikz`    `\usepackage [(usetikz=false)] {soup}`

Usually, `soup` will use TikZ to draw the soup grid and provide the optional highlighting of clues.

To disable this, and use a non-TikZ fallback (the `tabular` environment), pass the option `usetikz=false` when loading `soup`.

---

`highlight`    `\usepackage [(highlight=true)] {soup}`

As a puzzle generator, `soup` does not usually indicate the solution.

To have `soup` highlight the solutions, pass the option `highlight` (or `highlight=true`) when loading `soup`.

If TikZ is disabled, the solutions will be indicated with boldface letters. Note that if the puzzle is drawn in boldface, this will hide the highlighting.

---

`highlightcolor`    `\usepackage [(highlightcolor=color)] {soup}`

Specify the fill color to be used when highlighting solutions (TikZ only).

The default color is `orange`.

Color mixes are fine here, too: `green!50!white`.

---

`linecolor`    `\usepackage [(linecolor=color)] {soup}`

Specify the line color to be used when highlighting solutions (TikZ only).

The default color is `red`.

Color mixes are fine here, too: `green!20!black`.

## 1.2 Environments

---

<code>alphabetsoup</code>	<code>\begin{alphabetsoup} [<i>(width)</i>] [<i>(height)</i>] [<i>(font)</i>]</code>
<code>alphabetsoup*</code>	<code>\begin{alphabetsoup}* [<i>(width)</i>] [<i>(height)</i>] [<i>(font)</i>]</code>
<code>Alphabetsoup</code>	<code>\begin{Alphabetsoup} [<i>(width)</i>] [<i>(height)</i>] [<i>(font)</i>]</code>
<code>Alphabetsoup*</code>	<code>\begin{Alphabetsoup}* [<i>(width)</i>] [<i>(height)</i>] [<i>(font)</i>]</code>

---

An `alphabetsoup` environment will build a grid of letters using lowercase Latin a–z, weighted for their frequency in English words. The `Alphabetsoup` environment uses uppercase A–Z. (For other alphabets, use a custom `homemadesoup`.)

A list of clues will be included after the grid. Use the starred version to omit the list. (To include the list later, use `\listofclues`.)

If the `(height)` is omitted, the number of rows will be the same as the number of columns.

If the `(width)` is omitted, it will default to 20.

Therefore, with no parameters, a 20-by-20 grid of letters will be generated.

`(font)` can be optionally used to set the size of the letters in the soup (e.g., `\Large`, `\scriptsize`) or other font-related commands (e.g., `\sffamily`, `\itshape`)

---

<code>numbersoup</code>	<code>\begin{numbersoup} [<i>width</i>] [<i>height</i>] {<i>max</i>} [<i>min</i>] [<i>font</i>]</code>
<code>numbersoup*</code>	<code>\begin{numbersoup}* [<i>width</i>] [<i>height</i>] {<i>max</i>} [<i>min</i>] [<i>font</i>]</code>

---

The `numbersoup` environment follows `alphabetsoup` with two important differences:

- The grid is filled with numbers (not letters)
- Numbers are between `<min>` (or 0 if omitted) and `<max>`, inclusive.

The `<max>` must be specified.

---

<code>homemadesoup</code>	<code>\begin{homemadesoup} [<i>width</i>] [<i>height</i>] {<i>symbols</i>} [<i>font</i>]</code>
<code>homemadesoup*</code>	<code>\begin{homemadesoup}* [<i>width</i>] [<i>height</i>] {<i>symbols</i>} [<i>font</i>]</code>

---

Instead of filling with digits or letters, the soup will be filled randomly from the user-specified comma-separated list `<symbols>`

### 1.3 Macros

---

<code>\hideinsoup</code>	<code>\hideinsoup {<i>x</i>} {<i>y</i>} {<i>dir</i>} {<i>seq</i>} [<i>clue</i>]</code>
<code>\hideinsoup*</code>	

---

Generally, an `alphabetsoup` will have words hidden in it. Other soups will have appropriate clues hidden (e.g., a number series).

These are put in the soup with `\hideinsoup`.

If two words overlap, and the overlapping letters (or other symbols) are different, `soup` will issue a warning, and it will display *both* letters in the grid, separated by a slash.

If highlighting is enabled, `\hideinsoup` will call `\highlightinsoup`. Use the starred version, `\hideinsoup*` to avoid this behavior.

If `soup` was loaded with `usetikz=false`, the highlighting of hidden clues will be simple boldface. The starred version will have no effect on this.

---

<code>\highlightinsoup</code>	<code>\highlightinsoup {<i>x1</i>} {<i>y1</i>} {<i>x2</i>} {<i>y2</i>}</code>
-------------------------------	---

---

Highlights the word (or sequence of symbols) between  $(\langle x1 \rangle, \langle y1 \rangle)$  and  $(\langle x2 \rangle, \langle y2 \rangle)$ , where (1,1) is the top left of the soup grid, (2,1) is to the right of the top left, and (1,2) is the first symbol in the second row.

If `soup` was loaded with `usetikz=false`, this macro will have no effect.

---

`\listofclues`

`\listofclues [format]`

Displays a list of all clues for the current puzzle.

The optional *format* should use `\theclue` where the text of the clue should appear.

Must be used after all uses of `\hideinsoup` for the current soup. If included before `\end{...soup}`, the clues will appear *before* the soup. If included after `\end{...soup}`, then they will appear *after* the soup.

A typical use might be to display the clues as an enumerated list in columns:

```
\begin{alphabetsoup}*  
  ...  
\end{alphabetsoup}  
\begin{multicols}{3}  
  \begin{enumerate}  
    \listofclues[\item \theclue]  
  \end{enumerate}  
\end{multicols}
```

## 2 Implementation

### 2.1 Dependencies

```
1 \RequirePackage{xparse}  
2 \RequirePackage{expl3}  
3 \RequirePackage{l3keys2e}
```

### 2.2 Initialization and Parameter Handling

```
4 \ExplSyntaxOn  
5  
6 \msg_new:nnn{soup}{mismatch}{  
7   Clue~mismatch~at~#1.~Will~appear~as~#2/#3~in~the~soup.  
8 }  
9  
10 \bool_new:N \g_soup_use_tikz_bool  
11 \bool_gset_true:N \g_soup_use_tikz_bool  
12  
13 \bool_new:N \g_soup_highlight_bool  
14 \bool_gset_false:N \g_soup_highlight_bool  
15  
16 \tl_new:N \g_soup_highlight_color  
17 \tl_gset:Nn \g_soup_highlight_color {orange}  
18  
19 \tl_new:N \g_soup_line_color  
20 \tl_gset:Nn \g_soup_line_color {red}  
21  
22 \keys_define:nn { soup }{  
23   highlightcolor .initial:n      = orange,  
24   highlightcolor .value_required:n = true,  
25   highlightcolor .code:n         = \tl_set:Nn \g_soup_highlight_color {#1},  
26   linecolor      .initial:n      = red,  
27   linecolor      .value_required:n = true,  
28   linecolor      .code:n         = \tl_set:Nn \g_soup_line_color {#1},
```

```

29 highlight      .default:n      = true,
30 highlight      .bool_set:N      = \g_soup_highlight_bool,
31 usetikz        .default:n      = true,
32 usetikz        .bool_set:N      = \g_soup_use_tikz_bool,
33 }
34
35 \ProcessKeysPackageOptions{ soup }
36 \IfBooleanT \g_soup_use_tikz_bool {
37   \RequirePackage{tikz}
38 }
39 \clist_const:Nn \c_soup_Alphabet_clist {
40   A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z,
41   E,T,A,O,H,N,I,S,R,D,L,U,W,M,C,G,F,Y,P,V,K,B,J,
42   E,T,A,O,H,N,I,S,R,D,L,U,W,M,C,G,F,Y,P,V,K,B,
43   E,T,A,O,H,N,I,S,R,D,L,U,W,M,
44   E,T,A,O,H,N,I,S,
45   E,T,A,O,H,
46 }
47
48 \clist_const:Nn \c_soup_alphabet_clist {
49   a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,
50   e,t,a,o,h,n,i,s,r,d,l,u,w,m,c,g,f,y,p,v,k,b,j,
51   e,t,a,o,h,n,i,s,r,d,l,u,w,m,c,g,f,y,p,v,k,b,
52   e,t,a,o,h,n,i,s,r,d,l,u,w,m,
53   e,t,a,o,h,n,i,s,
54   e,t,a,o,h,
55 }
56
57 \prop_new:N \g_soup_data_prop
58 \seq_new:N \g_soup_clue_seq

```

## 2.3 Internal Functions

`\__soup_init:nn` Resets the storage in preparation for a new soup.

```

59 \cs_new:Nn \__soup_init:nn {
60   \clist_clear_new:N \g_soup_symbol_clist
61   \dim_gzero_new:N \g_soup_highlight_dim
62   \dim_gzero_new:N \g_soup_spacing_dim
63   \int_gzero_new:N \g_soup_columns_int
64   \int_gzero_new:N \g_soup_number_max_int
65   \int_gzero_new:N \g_soup_number_min_int
66   \int_gzero_new:N \g_soup_number_range_int
67   \int_gzero_new:N \g_soup_rows_int
68   \int_gzero_new:N \g_soup_symbol_count_int
69   \prop_clear_new:N \g_soup_data_prop
70   \seq_clear_new:N \g_soup_clue_seq
71   \seq_clear_new:N \g_soup_highlight_seq
72   \int_gset:Nn \g_soup_columns_int {#1}
73   \IfNoValueTF{#2} {
74     \int_gset:Nn \g_soup_rows_int {\g_soup_columns_int}
75   }{
76     \int_gset:Nn \g_soup_rows_int {#2}
77   }
78   \dim_gset:Nn \g_soup_spacing_dim {\textwidth / (\g_soup_columns_int + 1)}

```

```

79   \dim_gset:Nn \g_soup_highlight_dim {\g_soup_spacing_dim * 7 / 10}
80   \tl_clear_new:N \g_soup_font_tl
81   \tl_gset:Nn \g_soup_font_tl {\normalfont}
82 }

```

(End definition for `\_soup_init:nn`.)

`\_soup_random_int:nn` Returns a pseudo-random integer between #1 and #2.

[https://en.wikipedia.org/wiki/Lehmer\\_random\\_number\\_generator](https://en.wikipedia.org/wiki/Lehmer_random_number_generator)

```

83 \int_gzero_new:N \g__soup_random_previous_int
84 \int_gzero_new:N \g__soup_random_current_int
85 \cs_new:Nn \_soup_random_int:nn {
86   \int_compare:nNnT \g__soup_random_previous_int = 0 {
87     \int_gset:Nn \g__soup_random_previous_int {\time}
88   }
89   % A = 16807, Q = 127773 (M / A), R = 2836 (M % A), M = 2147483647 (2^31-1)
90   \int_zero_new:N \l__hi_int
91   \int_zero_new:N \l__lo_int
92   \int_set:Nn \l__hi_int {\g__soup_random_previous_int / 127773}
93   \int_set:Nn \l__lo_int {\int_mod:nn{\g__soup_random_previous_int}{127773}}
94   \int_gset:Nn \g__soup_random_previous_int {
95     16807 * \l__hi_int - 2836 * \l__lo_int
96   }
97   \int_compare:nNnT \g__soup_random_previous_int < 1 {
98     \int_gadd:Nn \g__soup_random_previous_int {2147483647}
99   }
100  \int_gset:Nn \g__soup_random_current_int {
101    #1 + \int_mod:nn{\g__soup_random_previous_int}{#2 - #1 + 1}
102  }
103 }

```

(End definition for `\_soup_random_int:nn`.)

`\_soup_draw_nodes:` Must be used inside a `tikzpicture` environment.

For every node pushed, now draw a node using either the previously set value or one now generated by the `getrand` macro.

```

104 \cs_new:Nn \_soup_draw_nodes: {
105   \int_step_variable:nnnNn {1} {1} {\g_soup_columns_int} \l_tmpb_int {
106     \int_step_variable:nnnNn {1} {1} {\g_soup_rows_int} \l_tmpc_int {
107       \exp_args:Nnx
108       \prop_get:NnNTF \g_soup_data_prop {
109         (\l_tmpb_int,\l_tmpc_int)
110       } \l_tmpa_tl {
111         \node
112           at (\l_tmpb_int,\l_tmpc_int)
113           {\l_tmpa_tl};
114       }{
115         \node
116           at (\l_tmpb_int,\l_tmpc_int)
117           {\_soup_show_random_symbol:};
118       }
119     }
120   }
121 }

```

(End definition for `\_soup\_draw\_nodes:`.)

`\_soup\_draw\_highlights:` Must be used inside a `tikzpicture` environment.  
For every previously stored highlight coords, now draw the lines.

```
122 \cs_new:Nn \_soup\_draw\_highlights: {
123   \seq_map_inline:Nn \g\_soup\_highlight\_seq {
124     \draw[
125       double=\g\_soup\_highlight\_color,
126       double~distance=\g\_soup\_highlight\_dim,
127       line~width=2pt,
128       color=\g\_soup\_line\_color,
129       opacity=0.4,
130       line~cap=round
131     ] ##1;
132   }
133 }
```

(End definition for `\_soup\_draw\_highlights:`.)

`\_soup\_draw\_soup\_tikz:` Do the actual work of drawing the soup

```
134 \cs_new:Nn \_soup\_draw\_soup\_tikz: {
135
136   \tikzset{
137     every~node/.style={
138       font=\g\_soup\_font\_tl,
139     },
140   }
141   \begin{tikzpicture}[
142     x=\g\_soup\_spacing\_dim,
143     y=-\g\_soup\_spacing\_dim,
144   ]
145     \draw[rounded~corners=6pt, use~as~bounding~box]
146       (0.5,0)
147       ++(0,0.5) rectangle +(\g\_soup\_columns\_int, \g\_soup\_rows\_int);
148     \_soup\_draw\_highlights:
149     \_soup\_draw\_nodes:
150   \end{tikzpicture}
151 }
```

(End definition for `\_soup\_draw\_soup\_tikz:`.)

`\_soup\_draw\_soup\_tabular:` Do the actual work of drawing the soup (as a table)

```
152 \cs_new:Nn \_soup\_draw\_soup\_tabular: {
153   \dim_zero_new:N \l\_soup\_colwidth\_dim
154   \exp_args:Nnx
155   \dim_set:Nn \l\_soup\_colwidth\_dim {\fp_to_dim:n {0.45 * \textwidth / (\g\_soup\_columns\_int
156
157   \dim_zero_new:N \l\_soup\_lineheight\_dim
158   \dim_set:Nn \l\_soup\_lineheight\_dim {2\l\_soup\_colwidth\_dim - \baselineskip}
159
160   \setlength{\tabcolsep}{\l\_soup\_colwidth\_dim}
161   \vspace{0.25\g\_soup\_spacing\_dim}\par
162   \noindent
163   \begin{tabular*}{\textwidth}{
```

```

164     @{\extracolsep{\fill}}
165     | *{\g_soup_columns_int}{c@{\hskip\l_soup_colwidth_dim}} |
166   }
167   \hline\hrule{Opt}{\g_soup_spacing_dim}
168   \int_step_inline:nnnn {1} {1} {\g_soup_rows_int } {
169     \int_gset:Nn \g_tmpa_int {##1}
170     \int_step_variable:nnnNn {1} {1} {\g_soup_columns_int} \l_tmpb_int {
171       \exp_args:Nnx
172       \prop_get:NnNTF \g_soup_data_prop {
173         (\l_tmpb_int,\the\g_tmpa_int)
174       } \l_tmpa_tl {
175         \g_soup_font_tl
176         \IfBooleanTF{\g_soup_highlight_bool}{
177           {\bfseries\l_tmpa_tl}
178         }{
179           \l_tmpa_tl
180         }
181       }{
182         \g_soup_font_tl\__soup_show_random_symbol:
183       }
184       \int_compare:nNnT \l_tmpb_int < \g_soup_columns_int {
185         &
186       }
187     }
188     \int_compare:nNnTF \g_tmpa_int < \g_soup_rows_int {
189       \l_soup_lineheight_dim
190     }{
191       \l_soup_lineheight_dim\hline\end{tabular*}
192     }
193   }
194 }

```

(End definition for `\__soup_draw_soup_tabular:`.)

`\__soup_show_random_symbol:` Called for every coordinate not defined by calls to `\hideinsoup`, this generates a random symbol—either a number from the `\g_soup_number_range_int` (if nonzero) or from the list of symbols in `\g_soup_symbol_clist` set by `homemadesoup`, `alphabetsoup`, and `Alphabetsoup`.

```

195 \cs_new:Nn \__soup_show_random_symbol: {
196   \int_compare:nNnTF \g_soup_symbol_count_int = 0 {
197     \__soup_random_int:nn {\g_soup_number_min_int}{\g_soup_number_max_int}
198     \the\g__soup_random_current_int
199   }{
200     \__soup_random_int:nn {1}{\g_soup_symbol_count_int}
201     \clist_item:Nn \g_soup_symbol_clist {\g__soup_random_current_int}
202   }
203 }

```

(End definition for `\__soup_show_random_symbol:`.)

## 2.4 User Document Functions

**`\listofclues`** Display the list of clues. The optional argument will be expanded with `\theclue` as each clue. The default is defined as `\theclue\par`.

```

204 \NewDocumentCommand \listofclues { +o } {
205   \tl_clear_new:N \theclue
206   \IfNoValueTF{#1}{
207     \tl_set:Nn \l_tmpa_tl {\theclue\par}
208   }{
209     \tl_set:Nn \l_tmpa_tl {#1}
210   }
211   \seq_map_variable:NNn \g_soup_clue_seq \theclue {
212     \l_tmpa_tl
213   }
214 }

```

(End definition for `\listofclues`. This function is documented on page 4.)

**`\highlightinsoup`** Given the coordinates of a word (expressed as `{x1}{y1}{x2}{y2}`), this will mark the word (or other sequence).

This is automatically called for every clue hidden via `\hideinsoup`.

This does nothing unless `highlight=true` was passed to the package.

```

215 \NewDocumentCommand \highlightinsoup { m m m m }{
216   \bool_if:NT \g_soup_highlight_bool {
217     \seq_gput_left:Nx \g_soup_highlight_seq {(#1, #2) -- (#3, #4)}
218   }
219 }

```

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

**`\hideinsoup`** Given a starting coordinate, a direction, a comma-separated list of symbols, and an optional clue, set the appropriate coordinates to these symbols.

`{x1}`, `{y1}`, `{direction}`, `{word}`, [`clue`]

The starred version will disable highlighting (if enabled) to allow setting parts of the soup that are outside actual answers.

If a clue is specified, insert it into the `\listofclues`

```

220 \NewDocumentCommand \hideinsoup { smmmo } {
221   \int_zero_new:N \l__soup_dx_int
222   \int_zero_new:N \l__soup_dy_int
223
224   \str_case:nn {#4} {
225     {left}{
226       \int_set:Nn \l__soup_dx_int {-1}
227       \int_set:Nn \l__soup_dy_int { 0}
228     }
229     {right}{
230       \int_set:Nn \l__soup_dx_int { 1}
231       \int_set:Nn \l__soup_dy_int { 0}
232     }
233     {up}{
234       \int_set:Nn \l__soup_dx_int { 0}
235       \int_set:Nn \l__soup_dy_int {-1}
236     }
237     {upleft}{
238       \int_set:Nn \l__soup_dx_int {-1}
239       \int_set:Nn \l__soup_dy_int {-1}
240     }
241     {upright}{

```

```

242         \int_set:Nn \l__soup_dx_int { 1}
243         \int_set:Nn \l__soup_dy_int {-1}
244     }
245     {down}{
246         \int_set:Nn \l__soup_dx_int { 0}
247         \int_set:Nn \l__soup_dy_int { 1}
248     }
249     {downleft}{
250         \int_set:Nn \l__soup_dx_int {-1}
251         \int_set:Nn \l__soup_dy_int { 1}
252     }
253     {downright}{
254         \int_set:Nn \l__soup_dx_int { 1}
255         \int_set:Nn \l__soup_dy_int { 1}
256     }
257 }
258
259 \clist_set:Nn \l__soup_clue_clist {#5}
260 \int_zero_new:N \l__soup_clue_count_int
261 \int_set:Nn \l__soup_clue_count_int {\clist_count:N \l__soup_clue_clist}
262
263 \int_zero_new:N \l__soup_cx_int
264 \int_zero_new:N \l__soup_cy_int
265 \tl_clear_new:N \l__soup_ci_tl
266 \tl_clear_new:N \l__soup_ch_tl
267 \tl_clear_new:N \l__soup_nn_tl
268
269 \int_step_variable:nnnNn {1} {1} {\l__soup_clue_count_int} \l__soup_ci_tl {
270     \int_set:Nn \l__soup_cx_int
271     {#2 + \l__soup_dx_int * (\l__soup_ci_tl - 1)}
272
273     \int_set:Nn \l__soup_cy_int
274     {#3 + \l__soup_dy_int * (\l__soup_ci_tl - 1)}
275
276     \exp_args:Nnx
277     \tl_set:Nn \l__soup_ch_tl
278     {\clist_item:Nn \l__soup_clue_clist {\l__soup_ci_tl}}
279
280     \exp_args:Nnx
281     \tl_set:Nn \l__soup_nn_tl
282     {(\the\l__soup_cx_int,\the\l__soup_cy_int)}
283
284     \exp_args:Nnx
285     \tl_set:Nn \l__soup_cv_tl
286     {\exp_args:Nno \prop_item:Nn \g_soup_data_prop \l__soup_nn_tl}
287
288     \str_if_empty:NTF \l__soup_cv_tl {
289         \exp_args:Nnx \prop_gput:Noo \g_soup_data_prop {
290             \l__soup_nn_tl
291         } {\l__soup_ch_tl}
292     }{
293         \str_if_eq:NNF \l__soup_cv_tl \l__soup_ch_tl {
294             \msg_warning:nnxxx{soup}{mismatch}{
295                 \l__soup_nn_tl

```

```

296         }{\l__soup_cv_tl}{\l__soup_ch_tl}
297
298         \tl_put_left:Nx \l__soup_ch_tl
299         {\l__soup_cv_tl/}
300
301         \exp_args:Nnx
302         \prop_gput:Noo \g_soup_data_prop {\l__soup_nn_tl}
303         {\l__soup_ch_tl}
304     }
305 }
306 }
307
308 \IfBooleanF{#1}{
309     \exp_args:Nnx
310     \int_set:Nn \l__soup_cx_int
311     {#2 + \l__soup_dx_int * (\l__soup_clue_count_int - 1)}
312
313     \exp_args:Nnx
314     \int_set:Nn \l__soup_cy_int
315     {#3 + \l__soup_dy_int * (\l__soup_clue_count_int - 1)}
316
317     \exp_args:Nnx
318     \tl_set:Nn \l__soup_nn_tl
319     {(\the\l__soup_cx_int,\the\l__soup_cy_int)}
320
321     \exp_args:Nnx
322     \seq_gput_left:Nx \g_soup_highlight_seq
323     {(#2, #3) -- \l__soup_nn_tl}
324 }
325 \IfNoValueF{#6}{
326     \seq_gput_left:No \g_soup_clue_seq {#6}
327 }
328 }

```

(End definition for `\hideinsoup` and `\hideinsoup*`. These functions are documented on page 3.)

## 2.5 Environments

**alphabetsoup** A soup environment where unspecified coordinates are fill with a–z  
**alphabetsoup\*** For something else, see the `homemadesoup` environment.

```

329 \NewDocumentEnvironment{alphabetsoup}{sO{15}oo }
330 {
331     \par\noindent
332     \__soup_init:nn {#2}{#3}
333     \IfBooleanTF{#1}{
334         \def\showlist{}
335     }{
336         \def\showlist{\par\vspace*{1em}\listofclues}
337     }
338     \IfNoValueF{#4}{
339         \tl_gset:Nn \g_soup_font_tl {#4}
340     }
341
342     \clist_gset_eq:NN \g_soup_symbol_clist

```

```

343     \c_soup_alphabet_clist
344
345     \int_gset:Nn \g_soup_symbol_count_int
346     {\clist_count:N \g_soup_symbol_clist}
347   }{
348     \IfBooleanTF \g_soup_use_tikz_bool {
349       \__soup_draw_soup_tikz:
350     }{
351       \__soup_draw_soup_tabular:
352     }
353     \showlist
354   }

```

(End definition for `alphabetsoup` and `alphabetsoup*`. These functions are documented on page 2.)

**Alphabetsoup** A soup environment where unspecified coordinates are A, B, C, D, E, F, G, H, I, J, K,  
**Alphabetsoup\*** L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z

For something else, see the `homemadesoup` environment.

```

355 \NewDocumentEnvironment{Alphabetsoup}{s0{15}oo }
356 {
357   \par\noindent
358   \__soup_init:nn {#2}{#3}
359   \IfBooleanTF{#1}{
360     \def\showlist{}
361   }{
362     \def\showlist{\par\vspace*{1em}\listofclues}
363   }
364   \IfNoValueF{#4}{
365     \tl_gset:Nn \g_soup_font_tl {#4}
366   }
367
368   \clist_gset_eq:NN \g_soup_symbol_clist
369   \c_soup_Alphabet_clist
370
371   \int_gset:Nn \g_soup_symbol_count_int
372   {\clist_count:N \g_soup_symbol_clist}
373 }{
374   \IfBooleanTF \g_soup_use_tikz_bool {
375     \__soup_draw_soup_tikz:
376   }{
377     \__soup_draw_soup_tabular:
378   }
379   \showlist
380 }

```

(End definition for `Alphabetsoup` and `Alphabetsoup*`. These functions are documented on page 2.)

**homemadesoup** The `homemadesoup` environment builds a soup from the user-supplied comma-separated  
**homemadesoup\*** list of symbols.

```

381 \NewDocumentEnvironment{homemadesoup}{s0{15}omo }
382 {
383   \par\noindent
384   \__soup_init:nn {#2}{#3}
385   \IfBooleanTF{#1}{

```

```

386     \def\showlist{}
387   }{
388     \def\showlist{\par\vspace*{1em}\listofclues}
389   }
390   \IfNoValueF{#5}{
391     \tl_gset:Nn \g_soup_font_tl {#5}
392   }
393
394   \clist_gset:Nn \g_soup_symbol_clist
395     {#4}
396
397   \int_gset:Nn \g_soup_symbol_count_int
398     {\clist_count:N \g_soup_symbol_clist}
399 }
400 {
401   \IfBooleanTF \g_soup_use_tikz_bool {
402     \__soup_draw_soup_tikz:
403   }{
404     \__soup_draw_soup_tabular:
405   }
406   \showlist
407 }

```

(End definition for `homemadesoup` and `homemadesoup*`. These functions are documented on page 3.)

**numbersoup** Sets up a soup with all unspecified coordinates displaying numbers.

**numbersoup\***

```

408 \NewDocumentEnvironment{numbersoup}{ s0{15}om0{0}o }
409 {
410   \par\noindent
411   \__soup_init:nn{#2}{#3}
412   \IfBooleanTF{#1}{
413     \def\showlist{}
414   }{
415     \def\showlist{\par\vspace*{1em}\listofclues}
416   }
417   \IfNoValueF{#6}{
418     \tl_gset:Nn \g_soup_font_tl {#6}
419   }
420
421   \int_gset:Nn \g_soup_number_max_int
422     {#4}
423
424   \int_gset:Nn \g_soup_number_min_int
425     {#5}
426
427   \int_gset:Nn \g_soup_number_range_int
428     {\g_soup_number_max_int - \g_soup_number_min_int}
429 }
430 {
431   \IfBooleanTF \g_soup_use_tikz_bool {
432     \__soup_draw_soup_tikz:
433   }{
434     \__soup_draw_soup_tabular:
435   }

```

```
436 \showlist
437 }
```

*(End definition for numbersoup and numbersoup\*. These functions are documented on page 3.)*

```
438 \ExplSyntaxOff
```

# Change History

v1.0		v1.0.2	
General: Initial version . . . . .	1	General: Update to work with changes to expl3 kernel. . . . .	1

# Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

<b>A</b>		<b>H</b>	
Alphabetsoup	2, <u>355</u>	\hideinsoup	3, <u>220</u>
alphabetsoup	2, <u>329</u>	\hideinsoup*	3, <u>220</u>
Alphabetsoup*	2, <u>355</u>	highlight	2
alphabetsoup*	2, <u>329</u>	highlightcolor	2
<b>B</b>		\highlightinsoup	3, <u>215</u>
\baselineskip	158	\hline	167, <u>191</u>
\begin{Alphabetsoup}	2	homemadesoup	3, <u>381</u>
\begin{alphabetsoup}	2	homemadesoup*	3, <u>381</u>
\begin{homemadesoup}	3	\hskip	165
\begin{numbersoup}	3	<b>I</b>	
\bfseries	177	\IfBooleanTF	36, 176, 308,
bool commands:			333, 348, 359, 374, 385, 401, 412, 431
\bool_gset_false:N	14	\IfNoValueTF	73, 206, 325, 338, 364, 390, 417
\bool_gset_true:N	11	int commands:	
\bool_if:NTF	216	\int_compare:nNnTF	86, 97, 184, 188, 196
\bool_new:N	10, 13	\int_gadd:Nn	98
<b>C</b>		\int_gset:Nn	72, 74, 76, 87, 94,
clist commands:			100, 169, 345, 371, 397, 421, 424, 427
\clist_clear_new:N	60	\int_gzero_new:N	
\clist_const:Nn	39, 48		63, 64, 65, 66, 67, 68, 83, 84
\clist_count:N	261, 346, 372, 398	\int_mod:nn	93, 101
\clist_gset:Nn	394	\int_set:Nn	92,
\clist_gset_eq:NN	342, 368		93, 226, 227, 230, 231, 234, 235,
\clist_item:Nn	201, 278		238, 239, 242, 243, 246, 247, 250,
\clist_set:Nn	259		251, 254, 255, 261, 270, 273, 310, 314
cs commands:		\int_step_inline:nnnn	168
\cs_new:Nn	59, 85, 104, 122, 134, 152, 195	\int_step_variable:mnnNn	
<b>D</b>			105, 106, 170, 269
dim commands:		\int_zero_new:N	
\dim_gset:Nn	78, 79		90, 91, 221, 222, 260, 263, 264
\dim_gzero_new:N	61, 62	\g_tmpa_int	169, 173, 188
\dim_set:Nn	155, 158	\l_tmpb_int	
\dim_zero_new:N	153, 157		105, 109, 112, 116, 170, 173, 184
<b>E</b>		\l_tmpc_int	106, 109, 112, 116
exp commands:		int internal commands:	
\exp_args:Nno	286	\l_hi_int	90, 92, 95
\exp_args:Nnx	107, 154, 171, 276,	\l_lo_int	91, 93, 95
	280, 284, 289, 301, 309, 313, 317, 321	<b>K</b>	
\extracolsep	164	keys commands:	
<b>F</b>		\keys_define:mn	22
\fill	164	<b>L</b>	
fp commands:		linecolor	2
\fp_to_dim:n	155	\listofclues	4, <u>204</u> , 336, 362, 388, 415

<b>M</b>	
msg commands:	
\msg_new:nnn	6
\msg_warning:nmnn	294
<b>N</b>	
\normalfont	81
numbersoup	3, 408
numbersoup*	3, 408
<b>P</b>	
prop commands:	
\prop_clear_new:N	69
\prop_get:NnNTF	108, 172
\prop_gput:Nnn	289, 302
\prop_item:Nn	286
\prop_new:N	57
<b>R</b>	
\rule	167
<b>S</b>	
seq commands:	
\seq_clear_new:N	70, 71
\seq_gput_left:Nn	217, 322, 326
\seq_map_inline:Nn	123
\seq_map_variable:NNn	211
\seq_new:N	58
\setlength	160
\showlist	334, 336, 353, 360, 362, 379, 386, 388, 406, 413, 415, 436
soup commands:	
\c_soup_Alphabet_clist	39, 369
\c_soup_alphabet_clist	48, 343
\g_soup_clue_seq	58, 70, 211, 326
\g_soup_columns_int	63, 72, 74, 78, 105, 147, 155, 165, 170, 184
\l_soup_colwidth_dim	153, 155, 158, 160, 165
\g_soup_data_prop	57, 69, 108, 172, 286, 289, 302
\g_soup_font_tl	80, 81, 138, 175, 182, 339, 365, 391, 418
\g_soup_highlight_bool	13, 14, 30, 176, 216
\g_soup_highlight_color	16, 17, 25, 125
\g_soup_highlight_dim	61, 79, 126
\g_soup_highlight_seq	71, 123, 217, 322
\g_soup_line_color	19, 20, 28, 128
\l_soup_lineheight_dim	157, 158, 189, 191
\g_soup_number_max_int	64, 197, 421, 428
\g_soup_number_min_int	65, 197, 424, 428
\g_soup_number_range_int	66, 427
\g_soup_rows_int	67, 74, 76, 106, 147, 168, 188
\g_soup_spacing_dim	62, 78, 79, 142, 143, 161, 167
\g_soup_symbol_clist	60, 201, 342, 346, 368, 372, 394, 398
\g_soup_symbol_count_int	68, 196, 200, 345, 371, 397
\g_soup_use_tikz_bool	10, 11, 32, 36, 348, 374, 401, 431
soup internal commands:	
\l__soup_ch_tl	266, 277, 291, 293, 296, 298, 303
\l__soup_ci_tl	265, 269, 271, 274, 278
\l__soup_clue_clist	259, 261, 278
\l__soup_clue_count_int	260, 261, 269, 311, 315
\l__soup_cv_tl	285, 288, 293, 296, 299
\l__soup_cx_int	263, 270, 282, 310, 319
\l__soup_cy_int	264, 273, 282, 314, 319
\__soup_draw_highlights:	122, 148
\__soup_draw_nodes:	104, 149
\__soup_draw_soup_tabular:	152, 351, 377, 404, 434
\__soup_draw_soup_tikz:	134, 349, 375, 402, 432
\l__soup_dx_int	221, 226, 230, 234, 238, 242, 246, 250, 254, 271, 311
\l__soup_dy_int	222, 227, 231, 235, 239, 243, 247, 251, 255, 274, 315
\__soup_init:mn	59, 332, 358, 384, 411
\l__soup_nn_tl	267, 281, 286, 290, 295, 302, 318, 323
\g__soup_random_current_int	84, 100, 198, 201
\__soup_random_int:mn	83, 197, 200
\g__soup_random_previous_int	83, 86, 87, 92, 93, 94, 97, 98, 101
\__soup_show_random_symbol:	117, 182, 195
str commands:	
\str_case:mn	224
\str_if_empty:NNTF	288
\str_if_eq:NNTF	293
<b>T</b>	
\tabcolsep	160
\textwidth	78, 155, 163
\the	173, 198, 282, 319
\theclue	205, 207, 211
\tikzset	136
\time	87

tl commands:	.. 25, 28, 207, 209, 277, 281, 285, 318
\tl_clear_new:N 80, 205, 265, 266, 267	\l_tmpa_tl ..... 110, 113, 174, 177, 179, 207, 209, 212
\tl_gset:Nn 17, 20, 81, 339, 365, 391, 418	
\tl_new:N ..... 16, 19	<b>U</b>
\tl_put_left:Nn ..... 298	\usepackage ..... 2, 2, 2, 2
\tl_set:Nn ..... 2	usetikz ..... 2