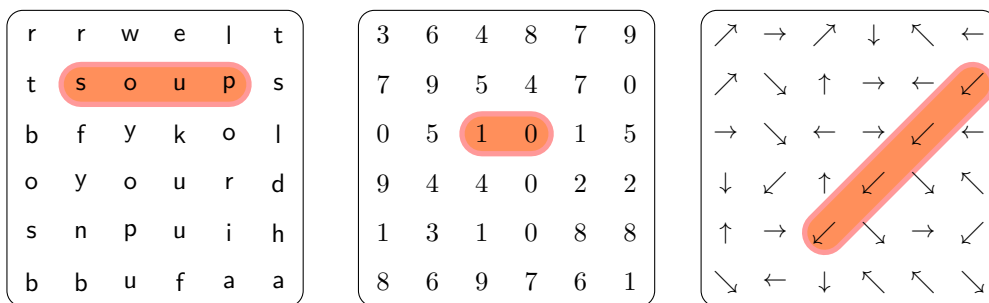


# The **soup** package <sup>\*</sup>

Thomas Simers <sup>†</sup>

Released 2017/01/15



## 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, last revised 2017/01/15.

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



# 1 User Guide

The `soup` interface is 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

---

---

<b>usetikz</b>	<code>\usepackage [<i>\usetikz=false</i>] {soup}</code>
----------------	---

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`.

---

---

<b>highlight</b>	<code>\usepackage [<i>\highlight=true</i>] {soup}</code>
------------------	--

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 the puzzle is drawn in boldface, this will hide the highlighting.

---

---

<b>highlightcolor</b>	<code>\usepackage [<i>\highlightcolor=color</i>] {soup}</code>
-----------------------	--

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`.

---

---

<b>linecolor</b>	<code>\usepackage [<i>\linecolor=color</i>] {soup}</code>
------------------	---

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>&lt;width&gt;</i>] [<i>&lt;height&gt;</i>] [<i>&lt;font&gt;</i>]</code>
<code>alphabetsoup*</code>	<code>\begin{alphabetsoup}* [<i>&lt;width&gt;</i>] [<i>&lt;height&gt;</i>] [<i>&lt;font&gt;</i>]</code>
<code>Alphabetsoup</code>	<code>\begin{Alphabetsoup} [<i>&lt;width&gt;</i>] [<i>&lt;height&gt;</i>] [<i>&lt;font&gt;</i>]</code>
<code>Alphabetsoup*</code>	<code>\begin{Alphabetsoup}* [<i>&lt;width&gt;</i>] [<i>&lt;height&gt;</i>] [<i>&lt;font&gt;</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>&lt;width&gt;</i>] [<i>&lt;height&gt;</i>] {<i>&lt;max&gt;</i>} [<i>&lt;min&gt;</i>] [<i>&lt;font&gt;</i>]</code>
<code>numbersoup*</code>	<code>\begin{numbersoup}* [<i>&lt;width&gt;</i>] [<i>&lt;height&gt;</i>] {<i>&lt;max&gt;</i>} [<i>&lt;min&gt;</i>] [<i>&lt;font&gt;</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>&lt;width&gt;</i>] [<i>&lt;height&gt;</i>] {<i>&lt;symbols&gt;</i>} [<i>&lt;font&gt;</i>]</code>
<code>homemadesoup*</code>	<code>\begin{homemadesoup}* [<i>&lt;width&gt;</i>] [<i>&lt;height&gt;</i>] {<i>&lt;symbols&gt;</i>} [<i>&lt;font&gt;</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>&lt;x&gt;</i>} {<i>&lt;y&gt;</i>} {<i>&lt;dir&gt;</i>} {<i>&lt;seq&gt;</i>} [<i>&lt;clue&gt;</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.



<hr/> <hr/>	<code>\highlightinsoup {⟨x1⟩} {⟨y1⟩} {⟨x2⟩} {⟨y2⟩}</code> <p>Highlights the word (or sequence of symbols) between <math>(\langle x1 \rangle, \langle y1 \rangle)</math> and <math>(\langle x2 \rangle, \langle y2 \rangle)</math>, 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.</p> <p>If <code>soup</code> was loaded with <code>usetikz=false</code>, this macro will have no effect.</p>
<hr/> <hr/>	<code>\listofclues [⟨format⟩]</code> <p>Displays a list of all clues for the current puzzle.</p> <p>The optional <math>\langle format \rangle</math> should use <code>\theclue</code> where the text of the clue should appear.</p> <p>Must be used after all uses of <code>\hideinsoup</code> for the current soup. If included before <code>\end{...soup}</code>, the clues will appear <i>before</i> the soup. If includes after <code>\end{...soup}</code>, then they will appear <i>after</i> the soup.</p> <p>A typical use might be to display the clues as an enumerated list in columns:</p> <pre> \begin{alphabetsoup}* ... \end{alphabetsoup} \begin{multicols}{3}   \begin{enumerate}     \listofclues[\item \theclue]   \end{enumerate} \end{multicols} </pre>

## 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:oo` Resets the storage in preparation for a new soup.

```

59 \cs_new:Nn \__soup_init:oo {
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:oo.)

\\_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     \tikzset{
136         every~node/.style={
137             font=\g_soup_font_tl,
138         },
139     }
140     \begin{tikzpicture}[
141
```



```

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_lineheight_dim
154     \dim_set:Nn \l_soup_lineheight_dim {\g_soup_spacing_dim - \baselineskip}
155
156     \vspace{0.25\g_soup_spacing_dim}\par
157     \noindent\fbbox{\parbox[c][
158         \g_soup_rows_int\g_soup_spacing_dim
159     ][c]{\g_soup_columns_int\g_soup_spacing_dim}{
160     \begin{tabular*}{
161         \g_soup_columns_int\g_soup_spacing_dim
162     }{
163         @{\extracolsep{\fill}}
164         *{\g_soup_columns_int}{c}
165     }
166     \int_step_inline:nnnn {1} {1} {\g_soup_rows_int} {
167         \int_gset:Nn \g_tmpa_int {##1}
168         \int_step_variable:nnnNn {1} {1} {\g_soup_columns_int} \l_tmpb_int {
169             \exp_args:Nnx
170             \prop_get:NnNTF \g_soup_data_prop {
171                 (\l_tmpb_int,\the\g_tmpa_int)
172             } \l_tmpa_tl {
173                 \g_soup_font_tl
174                 \IfBooleanTF{\g_soup_highlight_bool}{
175                     {\bfseries\l_tmpa_tl}
176                 }{
177                     \l_tmpa_tl
178                 }
179             }{
180                 \g_soup_font_tl\__soup_show_random_symbol:
181             }
182             \int_compare:nNnT \l_tmpb_int < \g_soup_columns_int {
183                 &
184             }
185         }
186     \int_compare:nNnTF \g_tmpa_int < \g_soup_rows_int {
187         \\\l_soup_lineheight_dim]

```



```

188         }{
189         }
190     }
191     \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 4.)

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

`\hideinsoup*`  $\{\langle x1 \rangle\}$ ,  $\{\langle y1 \rangle\}$ ,  $\{\langle direction \rangle\}$ ,  $\{\langle word \rangle\}$ ,  $[\langle clue \rangle]$

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 { smmmm } {
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**  
**alphabetsoup\***

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

```

329 \NewDocumentEnvironment{alphabetsoup}{s0{15}oo }
330 {
331   \par\noindent
332   \__soup_init:oo {#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 3.)

**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:oo {#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 3.)

**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:oo {#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:oo{#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

General: Initial version . . . . . 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.

A		F	
Alphabetsoup	3, <u>13</u>	\fbox	8
alphabetsoup	3, <u>12</u>	\fill	8
Alphabetsoup*	3, <u>13</u>	H	
alphabetsoup*	3, <u>12</u>	hi commands:	
B		\l__hi_int	6, 6, 6
\baselineskip	8	\hideinsoup	3, 3, <u>10</u> , <u>10</u>
\begin{Alphabetsoup}	3, 3	\hideinsoup*	3, <u>10</u>
\begin{alphabetsoup}	3, 3	highlight	2
\begin{homemadesoup}	3, 3	highlightcolor	2
\begin{numbersoup}	3, 3	\highlightinsoup	4, 4, <u>9</u> , 9
\bfseries	8	homemadesoup	3, <u>13</u>
bool commands:		homemadesoup*	3, <u>13</u>
\bool_gset_false:N	4	I	
\bool_gset_true:N	4	\IfBooleanF	12
\bool_if:NT	10	\IfBooleanT	5
\bool_new:N	4, 4	\IfBooleanTF	8, 12, 13, 13, 13, 13, 14, 14, 14
C		\IfNoValueF	12, 12, 13, 14, 14
clist commands:		\IfNoValueTF	6, 9
\clist_clear_new:N	5	int commands:	
\clist_const:Nn	5, 5	\int_compare:nNnT	6, 6, 8
\clist_count:N	11, 12, 13, 14	\int_compare:nNnTF	8, 9
\clist_gset:Nn	14	\int_gadd:Nn	6
\clist_gset_eq:NN	12, 13	\int_gset:Nn	6,
\clist_item:Nn	9, 11	6, 6, 6, 6, 6, 8, 12, 13, 14, 14, 14, 14	
\clist_set:Nn	11	\int_gzero_new:N	6, 6, 6, 6, 6, 6, 6, 6
cs commands:		\int_mod:nn	6, 6
\cs_new:Nn	5, 6, 7, 7, 7, 8, 9	\int_set:Nn	6, 6,
D		10, 10, 10, 10, 10, 10, 10, 10, 10,	
dim commands:		10, 10, 10, 10, 10, 10, 11, 11, 11, 12, 12	
\dim_gset:Nn	6, 6	\int_step_inline:nnnn	8
\dim_gzero_new:N	5, 5	\int_step_variable:nnnNn	7, 7, 8, 11
\dim_set:Nn	8	\int_zero_new:N	6, 6, 10, 10, 11, 11, 11
\dim_zero_new:N	8	K	
E		keys commands:	
exp commands:		\keys_define:nn	5
\exp_args:Nno	11	L	
\exp_args:Nnx	7,	linecolor	2
8, 11, 11, 11, 11, 11, 12, 12, 12, 12		\listofclues	4, 4, <u>9</u> , 9, 12, 13, 14, 14
\extracolsep	8	lo commands:	
		\l__lo_int	6, 6, 6



<b>M</b>		
msg commands:		
\msg_new:nnn	4	
\msg_warning:nnxxx	11	
<b>N</b>		
\normalfont	6	
numbersoup	3, 14	
numbersoup*	3, 14	
<b>P</b>		
\parbox	8	
prop commands:		
\prop_clear_new:N	6	
\prop_get:NnNTF	7, 8	
\prop_gput:Noo	11, 11	
\prop_item:Nn	11	
\prop_new:N	5	
<b>S</b>		
seq commands:		
\seq_clear_new:N	6, 6	
\seq_gput_left:No	12	
\seq_gput_left:Nx	10, 12	
\seq_map_inline:Nn	7	
\seq_map_variable:NNn	9	
\seq_new:N	5	
\showlist	12,	
	12, 13, 13, 13, 13, 13, 14, 14, 14, 14, 15	
soup commands:		
\c_soup_Alphabet_clist	5, 13	
\c_soup_alphabet_clist	5, 12	
\l_soup_ch_tl	11, 11, 11, 11, 11, 11, 11	
\l_soup_ci_tl	11, 11, 11, 11, 11	
\l_soup_clue_clist	11, 11, 11	
\l_soup_clue_count_int	11, 11, 11, 12, 12	
\g_soup_clue_seq	5, 6, 9, 12	
\g_soup_columns_int	6, 6, 6, 6, 7, 8, 8, 8, 8, 8, 8	
\l_soup_cv_tl	11, 11, 11, 11, 11	
\l_soup_cx_int	11, 11, 11, 12, 12	
\l_soup_cy_int	11, 11, 11, 12, 12	
\g_soup_data_prop	5, 6, 7, 8, 11, 11, 11	
\__soup_draw_highlights:	7, 7, 8	
\__soup_draw_nodes:	6, 7, 8	
\__soup_draw_soup_tabular:	8, 8, 13, 13, 14, 15	
\__soup_draw_soup_tikz:	7, 7, 13, 13, 14, 14	
\l_soup_dx_int	10,	
	10, 10, 10, 10, 10, 10, 10, 11, 12	
\l_soup_dy_int	10,	
	10, 10, 10, 10, 10, 10, 10, 11, 12	
\g_soup_font_tl	6, 6, 7, 8, 8, 12, 13, 14, 14	
\g_soup_highlight_bool	4, 4, 5, 8, 10	
\g_soup_highlight_color	4, 5, 5, 7	
\g_soup_highlight_dim	5, 6, 7	
\g_soup_highlight_seq	6, 7, 10, 12	
\__soup_init:oo	5, 5, 12, 13, 13, 14	
\g_soup_line_color	5, 5, 5, 7	
\l_soup_lineheight_dim	8, 8, 8	
\l_soup_nn_tl	11, 11, 11, 11, 11, 11, 12, 12	
\g_soup_number_max_int	6, 9, 14, 14	
\g_soup_number_min_int	6, 9, 14, 14	
\g_soup_number_range_int	6, 14	
\g_soup_random_current_int	6, 6, 9, 9	
\__soup_random_int:nn	6, 6, 9, 9	
\g_soup_random_previous_int	6, 6, 6, 6, 6, 6, 6, 6	
\g_soup_rows_int	6, 6, 6, 7, 8, 8, 8, 8	
\__soup_show_random_symbol:	7, 8, 9, 9	
\g_soup_spacing_dim	5, 6, 6, 8, 8, 8, 8, 8, 8	
\g_soup_symbol_clist	5, 9, 12, 12, 13, 13, 14, 14	
\g_soup_symbol_count_int	6, 9, 9, 12, 13, 14	
\g_soup_use_tikz_bool	4, 4, 5, 5, 13, 13, 14, 14	
str commands:		
\str_case:nn	10	
\str_if_empty:NTF	11	
\str_if_eq:NNF	11	
<b>T</b>		
\textwidth	6	
\the	8, 9, 11, 11, 12, 12	
\theclue	9, 9, 9	
\tikzset	7	
\time	6	
tl commands:		
\tl_clear_new:N	6, 9, 11, 11, 11	
\tl_gset:Nn	5, 5, 6, 12, 13, 14, 14	
\tl_new:N	4, 5	
\tl_put_left:Nx	11	



\tl_set:Nn . . . .	5, 5, 9, 9, 11, 11, 11, 12	tmpc commands:	
tmpa commands:		\l_tmpc_int . . . . .	7, 7, 7, 7
\g_tmpa_int . . . . .	8, 8, 8		
\l_tmpa_tl . . . . .	7, 7, 8, 8, 8, 9, 9, 9		U
tmpb commands:		\usepackage . . . . .	2, 2, 2, 2
\l_tmpb_int . . . . .	7, 7, 7, 7, 8, 8, 8	usetikz . . . . .	2