# Package 'spork'

May 9, 2026

**Type** Package

**Title** Generalized Label Formatting

**Version** 0.3.5

**Maintainer** Tim Bergsma <bergsmat@gmail.com>

**Description** The 'spork' syntax describes
label formatting concisely, supporting
mixed nesting of subscripts and superscripts
to arbitrary depth. It intends to be easy
to read and write in plain text, and easy
to convert to equivalent presentations
in 'plotmath', 'latex', and 'html'. Greek symbols
and a multiplication symbol are explicitly
supported. See ?as_spork and ?as_previews.

**License** GPL-3

**Encoding** UTF-8

**Imports** ggplot2, png, latexpdf, kableExtra

**RoxygenNote** 7.3.2

**Suggests** testthat (>= 2.1.0), magrittr, dplyr

**NeedsCompilation** no

**Author** Tim Bergsma [aut, cre]

**Repository** CRAN

**Date/Publication** 2024-10-12 02:30:02 UTC

## Contents

---

`as_html.spar` *Convert One Spork to Html*

---

## Description

Converts one spork to html. See description for `as_spork`. By default, unrecognized tokens are returned literally. However, Greek symbols and html metacharacters are escaped. See `htmlToken`.

## Usage

```
## S3 method for class 'spar'
as_html(
  x,
  newline = getOption("html_newline", "<br/>"),
  unrecognized = getOption("html_unrecognized", spork::htmlToken),
  token_open = getOption("html_token_open", ""),
  token_close = getOption("html_token_close", ""),
  math_open = getOption("html_math_open", ""),
  math_close = getOption("html_math_close", ""),
  label_open = getOption("html_label_open", ""),
  label_close = getOption("html_label_close", ""),
  ...
)
```

## Arguments

| | |
|---|---|
| `x` | spar |
| `newline` | value to replace `'\n'` |
| `unrecognized` | function to process unrecognized tokens: default `htmlToken` |
| `token_open, token_close` | these wrap text-like portions of the label; the defaults try to give upright characters (non-italic); also passed to `htmlToken` |
| `math_open, math_close` | these wrap math-like portions of the label; the defaults try to give upright characters (non-italic) which may not work for Greek symbols; also passed to `htmlToken` |
| `label_open, label_close` | these wrap the entire label |
| `...` | passed to `unrecognized`; see `htmlToken` |

## Details

Experimental support is implemented for the newline character (`'\n'`). Default behavior is to introduce linebreaks (<br/>) into the resulting html.

## Value

html

## See Also

Other interface: `as.expression.plotmath()`, `as_html.spork()`, `as_latex.spar()`, `as_latex.spork()`, `as_plotmath.spar()`, `as_plotmath.spork()`, `as_previews.spork()`, `as_spork.character()`, `htmlToken()`, `latexToken()`, `plotmathToken()`

Other html: `[.html()`, `[[.html()`, `as_html()`, `as_html.greek()`, `as_html.spork()`, `html2xml()`, `htmlToken()`

## Examples

```
library(magrittr)
'V_c./F' %>% as_spork %>% as_html
'AUC_ss' %>% as_spork %>% as_html
'C_max_ss' %>% as_spork %>% as_html
'var^eta_j' %>% as_spork %>% as_html
'& < % $ # \\_ { } ~ \\^ \\' %>% as_spork %>% as_html
'one joule (Omega) ~ 1 kg*m^2./s^2' %>% as_spork %>% as_html
'one joule (`Omega`) ~ 1 kg*m^2./s^2' %>% as_spork %>% as_html
'one joule (\\`Omega\\`) ~ 1 kg*m^2./s^2' %>% as_spork %>% as_html
```

---

# as_html.spork *Convert Spork to Html*

---

## Description

Converts spork to html. Vectorized version of `as_html.spar`.

## Usage

```
## S3 method for class 'spork'
as_html(x, ...)
```

## Arguments

| | |
|---|---|
| x | spork |
| ... | passed to `as_html.spar` |

## Value

html

## See Also

Other html: [.html(), [[.html(), as_html(), as_html.greek(), as_html.spar(), html2xml(), htmlToken()

Other spork: [.spork(), [[.spork(), as.list.spork(), as.png.spork(), as_latex.spork(), as_plotmath.spork(), as_previews.spork(), as_spar.default(), as_spar.spork(), as_spork(), as_spork.character(), as_spork.factor(), as_spork.spork(), ggplot.spork(), greek()

Other interface: as.expression.plotmath(), as_html.spar(), as_latex.spar(), as_latex.spork(), as_plotmath.spar(), as_plotmath.spork(), as_previews.spork(), as_spork.character(), htmlToken(), latexToken(), plotmathToken()

## Examples

```
x <- c(
  'V_c./F',
  'AUC_ss',
  'C_max_ss',
  'var^eta_j'
)
x <- as_spork(x)
as_html(x)
as_html(as_spork('gravitational force (kg\\.m/s^2.)'))
```

---

# as_latex.spar *Convert One Spork to Latex*

---

## Description

Converts one spork to latex. See description for as_spork. By default, unrecognized tokens are returned literally. However, Greek symbols and latex metacharacters are escaped. See latexToken.

## Usage

```
## S3 method for class 'spar'
as_latex(
  x,
  newline = getOption("latex_newline", "\n"),
  unrecognized = getOption("latex_unrecognized", spork::latexToken),
  token_open = getOption("latex_token_open", "\\textrm{"),
  token_close = getOption("latex_token_close", "}"),
  math_open = getOption("latex_math_open", "\\mathrm{"),
  math_close = getOption("latex_math_close", "}"),
  label_open = getOption("latex_label_open", "\\("),
  label_close = getOption("latex_label_close", "\\)"),
  enforce_math = getOption("latex_enforce_math", TRUE),
  script_size = getOption("latex_script_size", c("", "\\scriptsize ", "\\tiny ")),
  ...
)
```

## Arguments

| | |
|---|---|
| x | spar |
| newline | value to replace '\n' |
| unrecognized | function to process unrecognized tokens: default latexToken |
| token_open, token_close | these wrap text-like portions of the label; the defaults try to give upright characters (non-italic); also passed to latexToken |
| math_open, math_close | these wrap math-like portions of the label; the defaults try to give upright characters (non-italic) which may not work for Greek symbols; also passed to latexToken |
| label_open, label_close | these wrap the entire label; defaults invoke traditional math mode |
| enforce_math | whether to enforce math mode for nested expression: latexToken |
| script_size | three character values, one of which will be appended to token_open for unnested, nested, and multiply-nested contexts |
| ... | passed to unrecognized; see latexToken |

## Details

Experimental support is implemented for the newline character ('\n'). Default behavior is to introduce literal newline characters into the resulting tex. This may have no effect on the typeset result. It may be possible to achieve other effects by using non-default values of helper arguments and perhaps additional latex packages.

## Value

latex

## See Also

Other interface: as.expression.plotmath(), as_html.spar(), as_html.spork(), as_latex.spork(), as_plotmath.spar(), as_plotmath.spork(), as_previews.spork(), as_spork.character(), htmlToken(), latexToken(), plotmathToken()

Other latex: [.latex(), [[.latex(), as_latex(), as_latex.default(), as_latex.greek(), as_latex.latex(), as_latex.spork(), concatenate.latex(), latexToken()

## Examples

```
library(magrittr)
'V_c./F' %>% as_spork %>% as_latex
'AUC_ss' %>% as_spork %>% as_latex
'C_max_ss' %>% as_spork %>% as_latex
'var^eta_j' %>% as_spork %>% as_latex
'& % $ # \\_ { } ~ \\^ \\' %>% as_spork %>% as_latex
'one joule (Omega) ~ 1 kg*m^2./s^2' %>% as_spork %>% as_latex
'one joule (`Omega`) ~ 1 kg*m^2./s^2' %>% as_spork %>% as_latex
'one joule (\\`Omega\\`) ~ 1 kg*m^2./s^2' %>% as_spork %>% as_latex
```

as_latex.spork *Convert Spork to Latex*

## Description

Converts spork to latex. Vectorized version of as_latex.spar.

## Usage

```
## S3 method for class 'spork'
as_latex(x, ...)
```

## Arguments

| | |
|---|---|
| x | spork |
| ... | passed to as_latex.spar |

## Value

latex

## See Also

Other latex: [.latex(), [[.latex(), as_latex(), as_latex.default(), as_latex.greek(), as_latex.latex(), as_latex.spar(), concatenate.latex(), latexToken()

Other spork: [.spork(), [[.spork(), as.list.spork(), as.png.spork(), as_html.spork(), as_plotmath.spork(), as_previews.spork(), as_spar.default(), as_spar.spork(), as_spork(), as_spork.character(), as_spork.factor(), as_spork.spork(), ggplot.spork(), greek()

Other interface: as.expression.plotmath(), as_html.spar(), as_html.spork(), as_latex.spar(), as_plotmath.spar(), as_plotmath.spork(), as_previews.spork(), as_spork.character(), htmlToken(), latexToken(), plotmathToken()

## Examples

```
x <- c(
  'V_c./F',
  '\\nAUC_ss',
  'C_max_ss\\n',
  'var^eta_j\\nrecords'
)
x <- as_spork(x)
writeLines(as_latex(x))
x <- as_spork('gravitational force\\n (kg\\.m/s^2.)')
explicit(x)
as_latex(x)
```

`as_plotmath.spar` *Convert One Spork to Plotmath*

## Description

Converts one spork to plotmath. See description for `as_spork`. Unrecognized tokens are returned unmodified by default. Otherwise, backslashes and single quotes are escaped, and the result is wrapped in single quotes. See `plotmathToken`.

## Usage

```
## S3 method for class 'spar'
as_plotmath(
  x,
  unrecognized = getOption("plotmath_unrecognized", spork::plotmathToken),
  ...
)
```

## Arguments

| | |
|---|---|
| `x` | spar |
| `unrecognized` | function to process unrecognized tokens |
| `...` | passed to `unrecognized` |

## Details

Experimental support is implemented for the sequence "backslash n" (`'\n'`). It tries to break the expression at the point indicated, and stack the results. Active subscripts and superscripts are closed in advance, preventing these from breaking across lines.

## Value

character

## See Also

plotmathToken

Other interface: `as.expression.plotmath()`, `as_html.spar()`, `as_html.spork()`, `as_latex.spar()`, `as_latex.spork()`, `as_plotmath.spork()`, `as_previews.spork()`, `as_spork.character()`, `htmlToken()`, `latexToken()`, `plotmathToken()`

Other plotmath: `[.plotmath()`, `[[.plotmath()`, `as.expression.plotmath()`, `as.png.plotmath()`, `as_plotmath()`, `as_plotmath.greek()`, `as_plotmath.spork()`, `concatenate.plotmath()`, `ggplot.plotmath()`, `goodToken()`, `plotmathToken()`

Other spar: `as_spar()`, `as_spar.default()`, `as_spar.spork()`

## Examples

```
library(magrittr)
'V_c./F' %>% as_spork %>% as_plotmath
'AUC_ss' %>% as_spork %>% as_plotmath
'C_max_ss' %>% as_spork %>% as_plotmath
'var^eta_j' %>% as_spork %>% as_plotmath
'& % $ # \\_ { } ~ \\^ \\' %>% as_spork %>% as_plotmath
'one joule (Omega) ~ 1 kg*m^2./s^2' %>% as_spork %>% as_plotmath
'one joule (`Omega`) ~ 1 kg*m^2./s^2' %>% as_spork %>% as_plotmath
'one joule (\\`Omega\\`) ~ 1 kg*m^2./s^2' %>% as_spork %>% as_plotmath
```

---

# as_plotmath.spork *Convert Spork to Plotmath*

---

## Description

Converts spork to plotmath. See plotmath. Vectorized version of as_plotmath.spar.

## Usage

```
## S3 method for class 'spork'
as_plotmath(x, ...)
```

## Arguments

| | |
|---|---|
| x | spork |
| ... | passed to as_plotmath.spar |

## Value

plotmath

## See Also

Other plotmath: [.plotmath(), [[.plotmath(), as.expression.plotmath(), as.png.plotmath(), as_plotmath(), as_plotmath.greek(), as_plotmath.spar(), concatenate.plotmath(), ggplot.plotmath(), goodToken(), plotmathToken()

Other spork: [.spork(), [[.spork(), as.list.spork(), as.png.spork(), as_html.spork(), as_latex.spork(), as_previews.spork(), as_spar.default(), as_spar.spork(), as_spork(), as_spork.character(), as_spork.factor(), as_spork.spork(), ggplot.spork(), greek()

Other interface: as.expression.plotmath(), as_html.spar(), as_html.spork(), as_latex.spar(), as_latex.spork(), as_plotmath.spar(), as_previews.spork(), as_spork.character(), htmlToken(), latexToken(), plotmathToken()

## Examples

```
library(magrittr)
'V_c./F' %>% as_spork %>% as_plotmath
'AUC_ss' %>% as_spork %>% as_plotmath
'C_max_ss' %>% as_spork %>% as_plotmath
'var^eta_j' %>% as_spork %>% as_plotmath
'one joule (Omega) ~ 1 kg*m^2./s^2' %>% as_spork %>% as_plotmath
```

---

# as_previews.spork *Compare Previews of Spork*

---

## Description

Compares plotmath and latex previews of spork Generates png for both, and overlays latex above plotmath.

## Usage

```
## S3 method for class 'spork'
as_previews(x, wide = 70, long = 20, width = 3, height = 1, sleep = 2, ...)
```

## Arguments

| | |
|---|---|
| x | length-one spork |
| wide | width in mm of the latex image |
| long | length in mm of the latex image |
| width | width (default: inches) of the plotmath image |
| height | height (default: inches) of the plotmath image |
| sleep | how long to pause after html before latex/plotmath |
| ... | passed arguments |

## Value

invisible list of filepaths

## See Also

Other preview: as.png.plotmath(), as.png.spork(), as_preview(), as_preview.html(), as_preview.latex(), as_preview.plotmath(), as_previews(), as_previews.default(), ggplot.plotmath(), ggplot.spork()

Other interface: as.expression.plotmath(), as_html.spar(), as_html.spork(), as_latex.spar(), as_latex.spork(), as_plotmath.spar(), as_plotmath.spork(), as_spork.character(), htmlToken(), latexToken(), plotmathToken()

Other spork: [.spork(), [[.spork(), as.list.spork(), as.png.spork(), as_html.spork(), as_latex.spork(), as_plotmath.spork(), as_spar.default(), as_spar.spork(), as_spork(), as_spork.character(), as_spork.factor(), as_spork.spork(), ggplot.spork(), greek()

## Examples

```
library(magrittr)
specials <- '& % $ # \\_ { } ~ \\^ \\'

# specials %>% as_spork %>% as_previews
# specials %>% gsub(' ','',.) %>% as_spork %>% as_previews
# 'one joule (Omega) ~ 1 kg*m^2./s^2' %>% as_spork %>% as_previews

# disambiguation for plotmath and latex (see \code{\link[grDevices]{plotmath}}):

# 'epsilon.varepsilon' %>% as_spork %>% as_previews
# 'rho.varrho' %>% as_spork %>% as_previews
# 'Upsilon.Upsilon1' %>% as_spork %>% as_previews
# 'phi.phi1.varphi' %>% as_spork %>% as_previews
# 'sigma.sigma1.varsigma.stigma' %>% as_spork %>% as_previews
# 'theta.vartheta.theta1' %>% as_spork %>% as_previews
# 'omega.omega1.pi.varpi' %>% as_spork %>% as_previews
```

---

# as_spar.spork *Parse Spork*

---

## Description

Parses spork. Converts length-one character to vector of tokens. Explicit tokens include `*._^` and any of these escaped with backslash, e.g. `'\*'`. Backslash-n is an explicit token (`'\n'`). Backslash-backtick is an explicit token (`'\`'`). One or more consecutive whitespace characters are a single token, as are one or more consecutive octothorpes (#). Any string of characters delimited by one or more of the above is implicitly a token as well. As of version 0.2.6, supported names of Greek letters are tokens (see greek) possibly bounded by backticks (to be interpreted literally).

## Usage

```
## S3 method for class 'spork'
as_spar(x, ...)
```

## Arguments

| | |
|---|---|
| x | length-one character using spork syntax |
| ... | ignored arguments |

## Value

spar (character vector)

## See Also

Other spar: as_plotmath.spar(), as_spar(), as_spar.default()

Other spork: [.spork(), [[.spork(), as.list.spork(), as.png.spork(), as_html.spork(), as_latex.spork(), as_plotmath.spork(), as_previews.spork(), as_spar.default(), as_spork(), as_spork.character(), as_spork.factor(), as_spork.spork(), ggplot.spork(), greek()

## Examples

```
as_spar(as_spork('one joule (Omega) ~ 1 kg*m^2./s^2'))
as_spar(as_spork('one joule (`Omega`) ~ 1 kg*m^2./s^2'))
as_spar(as_spork('one joule (\\`Omega\\`) ~ 1 kg*m^2./s^2'))
```

---

# as_spork.character *Coerce Character to Spork*

---

## Description

Coerces character to class 'spork'. See description for as_spork.

## Usage

```
## S3 method for class 'character'
as_spork(x, ...)
```

## Arguments

| | |
|---|---|
| x | character |
| ... | ignored arguments |

## Value

spork

## See Also

Other spork: [.spork(), [[.spork(), as.list.spork(), as.png.spork(), as_html.spork(), as_latex.spork(), as_plotmath.spork(), as_previews.spork(), as_spar.default(), as_spar.spork(), as_spork(), as_spork.factor(), as_spork.spork(), ggplot.spork(), greek()

Other interface: as.expression.plotmath(), as_html.spar(), as_html.spork(), as_latex.spar(), as_latex.spork(), as_plotmath.spar(), as_plotmath.spork(), as_previews.spork(), htmlToken(), latexToken(), plotmathToken()

Other character: concatenate.character()

## Examples

```
as_spork('V_c./F')
```

`htmlToken` *Process Html Token*

## Description

Pre-processes a html token not recognized as spork. Escapes the common names for Greek letters and escapes html metacharacters.

## Usage

```
htmlToken(
  x,
  token_open = getOption("html_token_open", ""),
  token_close = getOption("html_token_close", ""),
  math_open = getOption("html_math_open", ""),
  math_close = getOption("html_math_close", ""),
  label_open = getOption("html_label_open", ""),
  label_close = getOption("html_label_close", ""),
  ...
)
```

## Arguments

| | |
|---|---|
| `x` | character |
| `token_open, token_close` | these wrap the entire token (used once); by default the token is text-like |
| `math_open, math_close` | these wrap math-like portions of the token; the defaults try to give upright characters (non-italic) which may not work for Greek symbols |
| `label_open, label_close` | these re-wrap math-like portions of the token |
| `...` | ignored arguments |

## Value

html

## See Also

Other html: `[.html()`, `[[.html()`, `as_html()`, `as_html.greek()`, `as_html.spar()`, `as_html.spork()`, `html2xml()`

Other interface: `as.expression.plotmath()`, `as_html.spar()`, `as_html.spork()`, `as_latex.spar()`, `as_latex.spork()`, `as_plotmath.spar()`, `as_plotmath.spork()`, `as_previews.spork()`, `as_spork.character()`, `latexToken()`, `plotmathToken()`

## Examples

```
htmlToken('foo')
htmlToken('alpha')
htmlToken('Alpha')
```

---

`latexToken` *Process Latex Token*

---

## Description

Pre-processes a latex token not recognized as spork. Escapes the common names for Greek letters and escapes latex metacharacters.

## Usage

```
latexToken(
  x,
  token_open = getOption("latex_token_open", "\\textrm{"),
  token_close = getOption("latex_token_close", "}"),
  math_open = getOption("latex_math_open", "\\mathrm{"),
  math_close = getOption("latex_math_close", "}"),
  label_open = getOption("latex_label_open", "\\("),
  label_close = getOption("latex_label_close", "\\)"),
  enforce_math = getOption("latex_enforce_math", TRUE),
  ...
)
```

## Arguments

| | |
|---|---|
| `x` | character |
| `token_open`, `token_close` | these wrap the entire token (used once); by default the token is text-like |
| `math_open`, `math_close` | these wrap math-like portions of the token; the defaults try to give upright characters (non-italic) which may not work for Greek symbols |
| `label_open`, `label_close` | these re-wrap math-like portions of the token if `enforce_math` is TRUE; defaults invoke traditional math mode |
| `enforce_math` | whether to enforce math mode for nested expression |
| `...` | ignored arguments |

## Value

latex

## See Also

Other latex: [.latex(), [[.latex(), as_latex(), as_latex.default(), as_latex.greek(), as_latex.latex(), as_latex.spar(), as_latex.spork(), concatenate.latex()

Other interface: as.expression.plotmath(), as_html.spar(), as_html.spork(), as_latex.spar(), as_latex.spork(), as_plotmath.spar(), as_plotmath.spork(), as_previews.spork(), as_spork.character(), htmlToken(), plotmathToken()

## Examples

```
latexToken('foo')
latexToken('alpha')
latexToken('Alpha')
```

---

# plotmathToken *Process Plotmath Token*

---

## Description

Processes a plotmath token. Escapes single-quotes and wraps in single-quotes. Also maps 'varepsilon' to 'epsilon', since plotmath has only the latter; likewise 'varrho' maps to 'rho' and 'varpi' maps to 'omega1'.

## Usage

```
plotmathToken(
  x,
  conditional = getOption("plotmath_conditional_quote", FALSE),
  unescape = getOption("plotmath_unescape", TRUE),
  ...
)
```

## Arguments

| | |
|---|---|
| x | (length-one) character |
| conditional | if true, return good tokens (parseable) unmodified; see goodToken |
| unescape | whether to escape (unrecognized) backslash |
| ... | ignored arguments |

## Value

plotmath

## See Also

Other plotmath: [.plotmath(), [[.plotmath(), as.expression.plotmath(), as.png.plotmath(), as_plotmath(), as_plotmath.greek(), as_plotmath.spar(), as_plotmath.spork(), concatenate.plotmath(), ggplot.plotmath(), goodToken()

Other interface: as.expression.plotmath(), as_html.spar(), as_html.spork(), as_latex.spar(), as_latex.spork(), as_plotmath.spar(), as_plotmath.spork(), as_previews.spork(), as_spork.character(), htmlToken(), latexToken()

## Examples

```
plotmathToken("can't")
plotmathToken("\\", unescape = TRUE)
plotmathToken("\\", unescape = FALSE)
plotmathToken("\n", conditional = TRUE)
plotmathToken("\n", conditional = FALSE)
plotmathToken('alpha')
plotmathToken('Alpha')
```

# Index