Package: regexplain (via r-universe)

July 12, 2024

Title Rstudio Addin to Explain, Test and Build Regular Expressions

Version 0.2.2.9000

Description A set of RStudio Addins to help interactively test and build regular expressions. Provides a Shiny gadget interface for interactively constructing the regular expression and viewing the results from common string-searching functions. The gadget interface includes a helpful regex syntax reference sheet and a library of common patterns.

License GPL-3

URL https://github.com/gadenbuie/regexplain

BugReports https://github.com/gadenbuie/regexplain/issues

Depends R (>= 3.2)

Imports dplyr, htmltools, knitr, miniUI, purrr, rlang, rstudioapi, shiny (>= 0.13), stringi, utils

Suggests jsonlite, rematch2, stringr, testthat

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Repository https://gadenbuie.r-universe.dev

RemoteUrl https://github.com/gadenbuie/regexplain

RemoteRef main

RemoteSha 326b316e1e7278d19c6f5a63cf1a8bb3ee416dc3

Contents

regex 2
regexplain_cheatsheet 2
regexplain_file 3
textAreaInputAlt 4
textInputCode 5
view_regex 6

Index **8**

`regex` *Extract matched groups from regexp*

Description

Extract matched groups from regexp

Usage

```
regex(
  text,
  pattern,
  ignore.case = FALSE,
  perl = FALSE,
  fixed = FALSE,
  useBytes = FALSE,
  global = TRUE
)
```

Arguments

`text`	Text to search
`pattern`	regexp
`ignore.case`	if `FALSE`, the pattern matching is *case sensitive* and if `TRUE`, case is ignored during matching.
`perl`	logical. Should Perl-compatible regexps be used?
`fixed`	logical. If `TRUE`, `pattern` is a string to be matched as is. Overrides all conflicting arguments.
`useBytes`	logical. If `TRUE` the matching is done byte-by-byte rather than character-by-character. See ‘Details’.
`global`	If `TRUE`, enables global pattern matching

`regexplain_cheatsheet` *Regex Cheatsheet Quick Reference*

Description

The function behind the RegExplain Cheatsheet addin. Opens a summary of regular expression syntax – the RegExplain cheatsheet – in an RStudio viewer pane.

Usage

```
regexplain_cheatsheet()
```

regexplain_file *RegExplain gadget*

Description

The function behind the RegExplain Selection and RegExplain File addins. Opens the RegExplain gadget interface in an RStudio viewer pane.

Usage

```
regexplain_file(pattern = NULL, start_page = "RegEx")

regexplain_gadget(
  text = NULL,
  pattern = NULL,
  start_page = if (is.null(text)) "Text" else "RegEx"
)

regexplain_web(text = NULL, pattern = NULL, start_page = "Text", ...)
```

Arguments

`pattern` Regular Expression to edit or visualize using RegExplain

`start_page` Open gadget to this tab, one of `"Text"`, `"RegEx"`, `"Output"`, or `"Help"`

`text` Text to explore in gadget (editable using interface)

`...` Arguments passed on to `shiny::shinyApp`

- `ui` The UI definition of the app (for example, a call to `fluidPage()` with nested controls)
- `server` A function with three parameters: `input`, `output`, and `session`. The function is called once for each session ensuring that each app is independent.
- `onStart` A function that will be called before the app is actually run. This is only needed for `shinyAppObj`, since in the `shinyAppDir` case, a `global.R` file can be used for this purpose.
- `options` Named options that should be passed to the `runApp` call (these can be any of the following: "port", "launch.browser", "host", "quiet", "display.mode" and "test.mode"). You can also specify `width` and `height` parameters which provide a hint to the embedding environment about the ideal height/width for the app.
- `uiPattern` A regular expression that will be applied to each `GET` request to determine whether the `ui` should be used to handle the request. Note that the entire request path must match the regular expression in order for the match to be considered successful.

enableBookmarking Can be one of "url", "server", or "disable". This is equivalent to calling the enableBookmarking() function just before calling shinyApp(). With the default value (NULL), the app will respect the setting from any previous calls to enableBookmarking(). See enableBookmarking() for more information.

Value

The regular expression built in the app is returned as a character string.

Functions

- regexplain_file: Opens file chooser to pick file, reads lines, returns first regexplain.addin.max_lines (default 100). Used in the "Regexplain File" regexplain_addin.
- regexplain_web: Launches the RegExplain gadget in a browser or an RStduio viewer pane.

Examples

```
## Not run:
regexplain_gadget(text = month.name, pattern = "(Ma|Ju)|(er)")
regexplain_web(text = month.name, pattern = "(Ma|Ju)|(er)")
regexplain_file()

## End(Not run)
```

textAreaInputAlt *Modified Text Area Input*

Description

Standard shiny::textAreaInput() with additional is_code parameter, added code font style for the input text and with autocomplete, autocorrect, autocapitalize and spellcheck set to off or false.

Usage

```
textAreaInputAlt(
  inputId,
  label,
  value = "",
  width = NULL,
  height = NULL,
  cols = NULL,
  rows = NULL,
  placeholder = NULL,
  resize = NULL,
  is_code = TRUE
)
```

Arguments

`inputId` The `input` slot that will be used to access the value.

`label` Display label for the control, or `NULL` for no label.

`value` Initial value.

`width` The width of the input, e.g. `'400px'`, or `'100%'`; see `validateCssUnit()`.

`height` The height of the input, e.g. `'400px'`, or `'100%'`; see `validateCssUnit()`.

`cols` Value of the visible character columns of the input, e.g. `80`. This argument will only take effect if there is not a CSS `width` rule defined for this element; such a rule could come from the `width` argument of this function or from a containing page layout such as `fluidPage()`.

`rows` The value of the visible character rows of the input, e.g. `6`. If the `height` argument is specified, `height` will take precedence in the browser's rendering.

`placeholder` A character string giving the user a hint as to what can be entered into the control. Internet Explorer 8 and 9 do not support this option.

`resize` Which directions the textarea box can be resized. Can be one of `"both"`, `"none"`, `"vertical"`, and `"horizontal"`. The default, `NULL`, will use the client browser's default setting for resizing textareas.

`is_code` Should the text input be considered verbatim code input?

See Also

Other modified shiny inputs: `textInputCode()`

`textInputCode` *Modified Text Input*

Description

Standard `shiny::textInput()` with additional `width` parameter, added code font style for the input text and with `autocomplete`, `autocorrect`, `autocapitalize` and `spellcheck` set to `off` or `false`.

Usage

```
textInputCode(
  inputId,
  label,
  value = "",
  width = NULL,
  placeholder = NULL,
  ...
)
```

Arguments

`inputId`	The `input` slot that will be used to access the value.
`label`	Display label for the control, or `NULL` for no label.
`value`	Initial value.
`width`	Width of `shiny-input-container` div.
`placeholder`	A character string giving the user a hint as to what can be entered into the control. Internet Explorer 8 and 9 do not support this option.
`...`	Extra elements to be included in the `input-group` div.

See Also

Other modified shiny inputs: `textAreaInputAlt()`

`view_regex` *View grouped regex results*

Description

View the result of the regular expression when applied to the given text. The default behavior renders the result as HTML and opens the file in the RStudio viewer pane. If `render` is `FALSE`, the HTML itself is returned. If the output is destined for a knitr document, set `knitr` to `TRUE`.

Usage

```
view_regex(
  text,
  pattern,
  ...,
  render = TRUE,
  escape = render,
  exact = escape,
  result_only = FALSE
)
```

Arguments

`text`	Text to search
`pattern`	Regex pattern to look for
`...`	Arguments passed on to `base::regexec` `ignore.case` if `FALSE`, the pattern matching is *case sensitive* and if `TRUE`, case is ignored during matching. `perl` logical. Should Perl-compatible regexps be used? `fixed` logical. If `TRUE`, `pattern` is a string to be matched as is. Overrides all conflicting arguments.

`useBytes` logical. If `TRUE` the matching is done byte-by-byte rather than character-by-character. See ‘Details’.

`render` Render results as HTML?

`escape` Escape HTML-related characters in `text`?

`exact` Should the regex pattern be displayed as entered by the user into R console or source (default)? When `TRUE`, regex is displayed with the double `\\\\` required for escaping backslashes in R. When `FALSE`, regex is displayed as interpreted by the regex engine (i.e. double `\\\\` as a single `\\`).

`result_only` Should only the result be displayed? If `FALSE`, then the colorized regular expression is also displayed in the output.

Examples

```
view_regex("example", "amp", render = FALSE)
```

Index

∗ **modified shiny inputs**
textAreaInputAlt, 4
textInputCode, 5

base::regexec, *6*

enableBookmarking(), *4*

fluidPage(), *5*

knitr, *6*

regex, 2
regexplain_addin, *4*
regexplain_cheatsheet, 2
regexplain_file, 3
regexplain_gadget *(*regexplain_file*)*, 3
regexplain_web *(*regexplain_file*)*, 3

shiny::shinyApp, *3*
shiny::textAreaInput(), *4*
shiny::textInput(), *5*

textAreaInputAlt, 4, *6*
textInputCode, *5*, 5

validateCssUnit(), *5*
view_regex, 6