Package: js4shiny (via r-universe)

August 22, 2024

Title Companion Package for JavaScript for Shiny Users

Version 0.0.28

Description Companion Package for JavaScript for Shiny Users.

License MIT + file LICENSE

URL https://github.com/gadenbuie/js4shiny

BugReports https://github.com/gadenbuie/js4shiny/issues

Depends R (>= 2.10)

Imports fs (>= 1.3.0), glue, htmltools, htmlwidgets, jsonlite, knitr, magrittr, purrr, rmarkdown (>= 1.11.0), shiny (>= 1.3.0), shinyAce, utils, yaml, zip

Suggests commonmark, golem, listviewer, r2d3, rstudioapi, servr, shinythemes, testthat (>= 2.1.0), usethis, xaringan

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

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

RemoteUrl https://github.com/gadenbuie/js4shiny

RemoteRef main

RemoteSha 8f30c1929b228e6e4b7bd092f169cbeb0083ae15

Contents

`first_sentences` *First Sentences of Books*

Description

First sentences of books, downloaded from `https://firstsentencesofbooks.tumblr.com/`.

Usage

```
first_sentences
```

Format

A data frame with 40 rows and 3 variables:

`quote` The first sentence of the book

`title` The title of the book

`author` The author of the book

`html_dependency_js4shiny`
js4shiny HTML Dependencies

Description

Include the various HTML JavaScript and CSS assets created for **js4shiny**.

Usage

```
html_dependency_js4shiny(
  redirectConsole = TRUE,
  jsonview = TRUE,
  stylize = "all",
  use_google_fonts = FALSE
)

html_dependency_redirectConsoleLog()

html_dependency_stylize(...)
```

Arguments

`redirectConsole`	Include JS and CSS assets to enable literate programming with JavaScript by redirecting `console.log()` to a chunk- specific output div.
`jsonview`	Include JS and CSS assets to enable tree view display of JSON objects for the **knitr** json engine.
`stylize`	One of "none", "all", "fonts", "variables", "table", "utility", "code", "pandoc-line-numbers" to include the CSS styles developed for **js4shiny**.
`use_google_fonts`	Should fonts hosted on Google Fonts be included? Default is `FALSE`, where only system fonts will be used.
`...`	Arguments passed to `html_dependency_js4shiny()`.

Functions

- `html_dependency_redirectConsoleLog`: Include just the console redirection dependencies.
- `html_dependency_stylize`: Include the full or partial **js4shiny** CSS styles.

See Also

Other js4shiny HTML dependencies: `html_setup()`

`html_document_js` *An HTML Document with Support for Literate JavaScript Programming*

Description

An R Markdown format for literate JavaScript programming. With default settings, each JavaScript chunk is run in its own environment and any output written with `console.log()` is inserted in the HTML document as the code runs. In this setting, the JavaScript is rendered directly in the browser at view time.

A similar effect can be achieved by using the `js_live = FALSE` chunk option to instead run the JavaScript code using node at compile time. In this setting, the results printed by the node process are captured and stored in the document, resulting in a non-dynamic output that captures the results of the JavaScript runtime code.

In both of the above settings, each code chunk is run separately. You can use the `js_redirect = FALSE` knitr chunk option to disable the `console.log()` redirect and use the standard JavaScript engine included in the **knitr** package. Logged statements will still be available in the browser's devolper tools console, as this engine is equivalent to having entered the JavaScript code directly into the HTML source within a `<script>` tag.

Usage

```
html_document_js(
  ...,
  theme = NULL,
  css = NULL,
  toc = FALSE,
  toc_depth = 3,
  mathjax = NULL,
  use_fontawesome = FALSE,
  use_google_fonts = FALSE,
  highlight = "haddock",
  fig_width = 10,
  fig_height = 7,
  fig_retina = 2,
  keep_md = FALSE,
  dev = "png",
  pandoc_args = NULL,
  extra_dependencies = NULL
)
```

Arguments

`...`	Additional function arguments to pass to the base R Markdown HTML output formatter `html_document_base`
`theme`	Ignored
`css`	One or more css files to include
`toc`	`TRUE` to include a table of contents in the output
`toc_depth`	Depth of headers to include in table of contents
`mathjax`	Include mathjax. The "default" option uses an https URL from a MathJax CDN. The "local" option uses a local version of MathJax (which is copied into the output directory). You can pass an alternate URL or pass `NULL` to exclude MathJax entirely.
`use_fontawesome`	Should FontAwesome be included? Default is `FALSE`.
`use_google_fonts`	Should fonts hosted on Google Fonts be included? Default is `FALSE`, where only system fonts will be used.

highlight	One of the pandoc highlight styles.
fig_width	Default width (in inches) for figures
fig_height	Default height (in inches) for figures
fig_retina	Scaling to perform for retina displays (defaults to 2, which currently works for all widely used retina displays). Set to NULL to prevent retina scaling. Note that this will always be NULL when keep_md is specified (this is because fig_retina relies on outputting HTML directly into the markdown document).
keep_md	Keep the markdown file generated by knitting.
dev	Graphics device to use for figure output (defaults to png)
pandoc_args	Additional command line options to pass to pandoc
extra_dependencies	Additional function arguments to pass to the base R Markdown HTML output formatter html_document_base

---

`html_document_js4shiny` *js4shiny Example Document*

Description

This document type is built on the `html_document_plain()`, but is configured to render the example documents created by the `repl()`. You may choose to render the solution or the example's initial state by setting the output option `version`.

Usage

```
html_document_js4shiny(version = c("solution", "initial"), ...)
```

Arguments

version	Which version of the example to render. One of `"solution"` (default) or `"initial"`.
...	Additional arguments passed to `html_document_plain()`

See Also

`html_document_plain()` `html_document_js()`

Examples

```
if (rmarkdown::pandoc_available("1.12.3")) {
  css_ex <- system.file(
    "examples", "css", "css-basics", "css-basics-appearance.Rmd",
    package = "js4shiny"
  )

  tmp_html_init <- tempfile("initial", fileext = ".html")
```

```
  tmp_html_sol <- tempfile("solution", fileext = ".html")

  tmp_html_init <- rmarkdown::render(
    input = css_ex,
    output_file = tmp_html_init,
    output_options = list(version = "initial"),
    quiet = TRUE
  )

  tmp_html_sol <- rmarkdown::render(
    input = css_ex,
    output_file = tmp_html_sol,
    output_options = list(version = "solution"),
    quiet = TRUE
  )
}

# View tmp_html_init/sol
# browseURL(tmp_html_init)
# browseURL(tmp_html_sol)
```

`html_document_plain` *Plain (Minimal) HTML Document*

Description

This RMarkdown output format provides a minimal HTML5 template and minimal features for including CSS and JavaScript files in the output source.

Usage

```
html_document_plain(
  ...,
  css = "normalize",
  script = NULL,
  highlight = "haddock",
  fig_width = 10,
  fig_height = 7,
  fig_retina = 2,
  keep_md = FALSE,
  dev = "png",
  pandoc_args = NULL,
  extra_dependencies = NULL
)

include_script(head = NULL, before = NULL, after = NULL)
```

Arguments

...	Additional function arguments to pass to the base R Markdown HTML output formatter html_document_base
css	A list of css files to include in the document's <head>. Include "normalize" in the list of css files to include normalize.css, which provides basic style resetting.
script	A list of .js files to include in the document. Use include_script() to choose where in the HTML the script should be sourced.
highlight	One of the pandoc highlight styles.
fig_width	Default width (in inches) for figures
fig_height	Default height (in inches) for figures
fig_retina	Scaling to perform for retina displays (defaults to 2, which currently works for all widely used retina displays). Set to NULL to prevent retina scaling. Note that this will always be NULL when keep_md is specified (this is because fig_retina relies on outputting HTML directly into the markdown document).
keep_md	Keep the markdown file generated by knitting.
dev	Graphics device to use for figure output (defaults to png)
pandoc_args	Additional command line options to pass to pandoc
extra_dependencies	Additional function arguments to pass to the base R Markdown HTML output formatter html_document_base
head, before, after	A character vector of source files, each to be included in the <head>, or *before* or *after* the HTML content in <body>.

Functions

- include_script: Helper function for including JS scripts

---

html_setup *Load js4shiny HTML assets and knitr dependencies*

Description

Overrides the JavaScript knitr engine, registers knitr output hooks, and declares the JS and CSS dependencies that are required to enable the literate JavaScript code chunks inside R Markdown formats that write to HTML.

Usage

```
html_setup(stylize = "none")

html_setup_blogdown(
  stylize = c("fonts", "variables", "table", "code", "utility")
)
```

Arguments

stylize	One of "none", "all", "fonts", "variables", "table", "utility", "code", "pandoc-line-numbers" to include the CSS styles developed for **js4shiny**.

Functions

- `html_setup_blogdown`: A blogdown-specific HTML setup that includes styles for `<pre>` code blocks, tables, some utility functions, and the CSS variables declaring the **js4shiny** colors.

See Also

Other js4shiny HTML dependencies: `html_dependency_js4shiny()`

`js4shiny_rmd` *Create a new js4shiny HTML document*

Description

Opens or creates an R Markdown document using the **js4shiny** html document templates.

Usage

```
js4shiny_rmd(
  type = c("plain", "js"),
  full_template = FALSE,
  path = NULL,
  overwrite = FALSE
)
```

Arguments

type	One of `"plain"` for `html_document_plain()` or `"js"` for `html_document_js()`.
full_template	Include the full R Markdown template document. Default is `FALSE`.
path	If `NULL`, an R Markdown document is opened in a new RStudio editor. If a path is given, a file is created and opened if in RStudio.
overwrite	If `TRUE`, will overwrite `path` if it exists.

See Also

`html_document_plain()`, `html_document_js()`

Examples

```
tmpfile <- tempfile(fileext = ".Rmd")
js4shiny_rmd(type = "plain", full_template = TRUE, path = tmpfile)
js4shiny_rmd(type = "plain", path = tmpfile, overwrite = TRUE)
```

knitr_js_engine *A JavaScript Engine for knitr*

Description

A JavaScript Engine for knitr

Usage

```
knitr_js_engine()
```

launch_shiny_in *Choose Launch Location for Shiny Apps*

Description

This function sets the `shiny.launch.browser` option to launch Shiny apps in an `"external"` browser, the RStudio viewer `"pane"`, or a new `"window"` in RStudio.

Usage

```
launch_shiny_in(where = NULL)
```

Arguments

`where`	One of `"external"`, `"pane"`, or `"window"`.

lint_js_addin *Lint and Fix JavaScript file with StandardJS*

Description

This addin lints and fixes selected JavaScript code or the currently open file in RStudio. The addin can be helpful for linting JavaScript code embedded in R Markdown or Shiny apps, in addition to linting whole JavaScript files. The underlying functions are not exported from **js4shiny**. If you want to programmatically lint multiple files, it would be better to use `npm` scripts or another JavaScript task running system to lint your files.

Installing StandardJS

standardjs is a style guide, code linter, and beautifier in one. It is also a command line tool (`standard`) for automatically formatting JavaScript code in the JavaScript Standard Style. The command line tool will also alert users to common style and programmer errors.

Using `standard` and this addin requires that `node`, `npm`, and `standard` be installed on your system. To install `node` and `npm`, you need to install Node.js (they come together). Follow the instructions from Node.js to install these tools. Confirm that your installation was successful by running `npm -v` in a new terminal session. Once `npm` is available, install `standard` globally by running this command in the terminal.

```
npm install standard --global
```

References

https://standardjs.com/

`live_preview` *Serve a Live Preview*

Description

Opens a live preview of the files in a directory. The live preview server automatically renders R Markdown files when they are saved, and the preview is refreshed whenever R Markdown files or supporting files, such as `.js`, `.css`, `.htm`, `.html`, `.sass`, or `.scss` files, are updated. This functionality requires the **servr** package.

Usage

```
live_preview(
  path = getwd(),
  update_pattern = "[.](js|css|[Rr]?[Mm][Dd]|html?|s[ca]ss)$",
  ...,
  render_quietly = getOption("js4shiny.live_preview.quiet", TRUE),
  external = FALSE
)

live_preview_stop(which = NULL)
```

Arguments

`path`	The path for the directory or file to preview. If the path given is an R Markdown document or HTML document, the HTML version of that file will be opened directly, otherwise the directory containing the file will be served.
`update_pattern`	Update the live preview when files matching this pattern are updated. By default, updating files with the following extensions will update the preview: `.Rmd` (case insensitive), `.html`, `.htm`, `.js`, `.css`, `.sass`, `.scss`.

`...` Arguments passed on to `servr::httw`

`dir` The root directory to serve.

`watch` a directory under which `httw()` is to watch for changes; if it is a relative path, it is relative to the `dir` argument

`pattern` a regular expression passed to `list.files()` to determine the files to watch

`all_files` whether to watch all files including the hidden files

`handler` a function to be called every time any files are changed or added under the directory; its argument is a character vector of the filenames of the files modified or added

`render_quietly` If `TRUE` (default), the output from `rmarkdown::render()` will not be shown. Set to `FALSE` for debugging. You can set the default value with a global option: `options(js4shiny.live_preview.quiet = FALSE)`

`external` Should the live preview be opened in an external browser? The default is `FALSE` and the preivew is opened in the RStudio viewer pane (if launched inside RStudio).

`which` A integer vector of the server IDs; by default, IDs of all existing servers in the current R session obtained from `daemon_list()`, i.e., all daemon servers will be stopped by default.

Value

Invisibly returns the `servr::httw()` object, so that you can manually stop the server with the `$stop_server()` method.

Functions

- `live_preview_stop`: Stop the live preview background daemons. See `servr::daemon_list()` for more information.

RStudio Addins

There are three Live Preview addins provided by **js4shiny**. **Live Preview** and **Live Preview (External)** open a live preview of the directory of the currently open document, if possible at the current HTML document corresponding to the open document. The external preview addin automatically opens the preview in your web browser, otherwise the preview is opened in the RStudio Viewer pane.

To stop the live server, you can call `servr::daemon_stop()` or `live_preview_stop()`, which will stop all bakground **servr** daemons, or you can use the **Live Preview Stop** addin.

Examples

```
if (interactive()) {

tmp_dir <- tempfile("live-preview")
dir.create(tmp_dir)
tmp_rmd <- file.path(tmp_dir, "js4shiny-plain.Rmd")
```

```
# Create a new js4shiny plain HTML document. If interactive
# and in RStudio, this file will open and you can use the
# addins to launch the live preview
js4shiny_rmd("js", full_template = TRUE, path = tmp_rmd)

srvr <- live_preview(tmp_rmd)

# Stop all background servers with either of the following
# live_preview_stop()
# servr::daemon_stop()
#
# Or if you've saved the return value from live_preview()
# srvr$stop_server()
}
```

`mdn_search` *Search the Mozilla Developers Network*

Description

Searches the MDN Web Docs for a search term. Includes an RStudio addin for quick and seamless searching: highlight a term in the source code and run the addin to search MDN. If no text is highlighted, the addin opens a Shiny gadget for you to enter your search term.

Usage

```
mdn_search(
  term,
  browse = TRUE,
  topics = c("js", "api", "css", "html", "svg"),
  locale = "en-US"
)

mdn_gadget(browse = TRUE, locale = NULL)
```

Arguments

`term`	Search term
`browse`	Should the search results be opened in a browser window? If not, the URL is returned instead.
`topics`	A selection of topics to search, choosing from `"js"`, `"api"`, `"css"`, `"html"`, `"svg"`, `"mobile"`, `"canvas"`, `"webdev"`, and `"standards"`.
`locale`	The locale string for the search query. Default is `"en-US"`, or set to `NULL` to search MDN without specifying a locale.

Value

The search URL

Functions

- `mdn_gadget`: A Shiny gadget for searching the MDN docs.

`register_knitr` *Register js4shiny knitr components*

Description

Register the js4shiny knitr JavaScript engine or the output hooks. Generally, you will not need to use these. Instead, see `html_document_js()` or `html_setup()` for methods that cover most use-cases.

Usage

```
register_knitr_output_hooks(set = TRUE, chunk_hook = NULL)

register_knitr_js_engine(set = TRUE)
```

Arguments

`set`	If `FALSE` the output hook or JS engine are returned rather than setting via knitr directly.
`chunk_hook`	Chunk hook to be applied *after* the js4shiny chunk hook is applied to the chunk output. If `NULL`, then the current chunk hook is used. Only applies when `set = TRUE`.

`repl_example` *REPL for live JS, CSS, and HTML development*

Description

Launches an interactive Shiny app for live editing of frontend JavaScript, CSS, and HTML/Markdown/R Markdown. The app allows users to write JS, CSS and HTML, preview the final product, observe the JavaScript console (specifically items printed to the console via `console.log()`), and download a zip file containing the source files.

Usage

```
repl_example(example = NULL)

repl(
  example = NULL,
  js_repl_only = FALSE,
  theme_app = NULL,
  theme_editor = "textmate",
  autocomplete = c("css", "html"),
  render_dir = NULL,
  options = list(),
  ...
)

repl_js(..., render_dir = NULL)
```

Arguments

example	The short name of the exercise or example, e.g. `ride-share-fares`. Alternatively, the path to a folder containing examples or the path to an example file directly. `repl_example(example = NULL)` opens an interactive browser to select an example, otherwise `repl()` and `repl_js()` will open with blank editors.
js_repl_only	When `TRUE`, the app is simplified to contain only a JavaScript source editor and a console output. `repl_js()` is an alias to launch `repl()` with `js_repl_only = TRUE`.
theme_app	The theme of the app, using **shinythemes**. See `shinythemes::shinytheme()` for a list of valid themes.
theme_editor	The theme of the **shinyAce** source code editors. See `shinyAce::getAceThemes()` for a list of valid themes.
autocomplete	Ace Editor language modes for which autocomplete will be enabled. One or more of `"js"`, `"css"`, or `"html"`. By default autocomplete is enabled in all but the JavaScript mode. "Disabling" autocomplete here actually doesn't mean disabling all together. Autocomplete will still be available by pressing `Ctrl + space`.
render_dir	Where to render temporary files, defaults to `tempdir()`
options	Options passed to `shiny::runApp()`.
...	Arguments passed from `repl_js()` to `repl()` or from `repl()` to `shiny::shinyApp()`.

Value

A shiny app

Functions

- `repl_example`: Launch a **js4shiny** exercise or example using the example slug, or the full filename. If none provided, `repl_example()` launches an interactive example browser.

Examples for js4shiny workshop

The app was developed for the **js4shiny** rstudio::conf workshop and can be used to load examples for practicing and learning JavaScript and web development concepts.

`snippets_install` *Install js4shiny snippets*

Description

This function installs a set of R, HTML, JavaScript and CSS snippets that are helpful when developing Shiny apps and doing web development work in RStudio. By default, the snippets are installed where RStudio will find them. If you haven't previously installed snippets to RStudio, these snippets will mask some of the built-in snippets that ship with RStudio.

Usage

```
snippets_install(install_path = NULL, update = TRUE)
```

Arguments

`install_path`	Where should the snippets be installed? If `NULL`, the snippets will install to a default path based on the current version of RStudio.
`update`	Should existing snippets be updated in place if there are any conflicts? Default is yes (`TRUE`). Otherwise, new snippets are appended to the end of the existing file, ensuring that you can recover your previous snippets by editing the snippets file.

Updating Existing Snippets

If you already have snippets installed, you can you can have the installed snippets update the existing snippets in place with `update = TRUE`. Or you can append the new snippets to the existing snippets files with `update = FALSE`. This option is desirable if you want to make sure that no snippets are overwritten. The newer snippets will mask older snippets, but no data will be lost.

Examples

```
snip_tmp <- tempfile("snippets")
dir.create(snip_tmp)
snippets_install(snip_tmp)
```

`us_cities_ranked` *125 US Cities Ranked, 2019*

Description

A ranking by U.S. news of the 125 most populous US metro areas to find the best places to live.

Usage

```
us_cities_ranked
```

Format

A data frame with 125 rows and 26 variables:

`us_news_rank` integer. Ranking by U.S. News
`city` character. City name
`state` character. State name, abbreviated
`state_full` character. State name, full
`metro_population` double. Population of meto area
`average_annual_salary` double. Average annual salary
`avg_temp_high_f` double. Average high temperature in ºF
`avg_temp_low_f` double. Average low temperature in ºF
`median_age` double. Median population age
`median_home_price` double. Median home price
`avg_annual_rainfall_in` double. Average annual rainfall in inches
`unemployment_rate` double. Unemployment rate
`median_monthly_rent` double. Median monthly rent cost
`avg_commute_time_mins` double. Average commute times in minutes
`percent_single` double. Percent of metro population that is single
`total_students` double. Total number of students
`total_teachers` double. Total number of teaches
`violent_crime` double. Crime rates per 100,000 people
`property_crime` double. Crime rates per 100,000 people
`link` character. Link to url on usanews.com
`lat` double. Latitude of metro area
`lon` double. Longitude of metro area
`lat_min` double. Latitude minimum bounding box of metro area
`lat_max` double. Latitude maximum bounding box of metro area
`lon_min` double. Longitude minimum bounding box of metro area
`lon_max` double. Longitude maximum bounding box of metro area

References

https://data.world/dataremixed/125-us-cities-ranked-2019, https://realestate.usnews.com/places/rankings/best-places-to-live

Index