Package: shinyjster (via r-universe)

September 20, 2024

Title Shiny JavaScript Testing

Version 0.0.0.9010

Description Run JavaScript testing on Shiny applications. This will act as a user clicking within the browser. Methods have been added to unify how testing is done.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Imports rmarkdown (>= 2.0.6), processx, htmltools, shiny, later, tibble, testthat, ellipsis

Suggests spelling, httpuv, callr, parallel

Language en-US

Roxygen list(markdown = TRUE)

Repository https://posit-dev-shinycoreci.r-universe.dev

RemoteUrl https://github.com/schloerke/shinyjster

RemoteRef HEAD

RemoteSha 865ea25f6d81296362da2e44b1a9318405c9a6a6

Contents

`assert_jster` *Assert jster app success*

Description

Verify that all jster apps tested successfully

Usage

```
assert_jster(dt)
```

Arguments

`dt`	`data.frame()` of information returned from `run_jster()` or `run_jster_apps()`

`js_script` *JavaScript helper*

Description

Wraps supplied text in an `htmltools::tags$script` call after turning it into `JS` code.

Usage

```
js_script(...)
```

Arguments

`...`	JavaScript text to be put in a script.

`run_jster` *Run an application with shinyjster enabled*

Description

Run an application with shinyjster enabled

Usage

```
run_jster(
  appDir,
  port = 8000,
  host = "127.0.0.1",
  browser = getOption("browser")
)

run_jster_apps(
  apps,
  type = c("serial", "callr", "lapply"),
  cores = parallel::detectCores(),
  port = NULL,
  host = "127.0.0.1",
  browser = getOption("browser")
)
```

Arguments

`appDir`	The application to run. Should be one of the following: • A directory containing `server.R`, plus, either `ui.R` or a `www` directory that contains the file `index.html`. • A directory containing `app.R`. • An `.R` file containing a Shiny application, ending with an expression that produces a Shiny app object. • A list with `ui` and `server` components. • A Shiny app object created by `shinyApp()`.
`port`	The TCP port that the application should listen on. If the `port` is not specified, and the `shiny.port` option is set (with `options(shiny.port = XX)`), then that port will be used. Otherwise, use a random port between 3000:8000, excluding ports that are blocked by Google Chrome for being considered unsafe: 3659, 4045, 5060, 5061, 6000, 6566, 6665:6669 and 6697. Up to twenty random ports will be tried.
`host`	The IPv4 address that the application should listen on. Defaults to the `shiny.host` option, if set, or `"127.0.0.1"` if not. See Details.

browser	a non-empty character string giving the name of the program to be used as the HTML browser. It should be in the PATH, or a full path specified. Alternatively, an R function to be called to invoke the browser. Under Windows NULL is also allowed (and is the default), and implies that the file association mechanism will be used.
apps	Vector of appDir values
type	Single value to determine how applications are executed. 'serial' Runs apps one after another using lapply. port will be random for each app unless specified. 'callr' Runs apps using callr::r_bg using cores cores. port will be random for each app to allow concurrent execution. 'lapply' Runs apps in succession using lapply. port will be random for each app unless specified.
cores	Number of cores (if needed) to execute on.

Functions

- run_jster(): Run a single shiny application with shinyjster enabled
- run_jster_apps(): Run a set of Shiny applications with shinyjster enabled

---

selenium_chrome *Selenium browsers*

Description

Opens a selenium driven browser and waits until shinyjster is finished.

Usage

```
selenium_chrome(
  timeout = 2 * 60,
  dimensions = "1200x1200",
  headless = !interactive(),
  verbose = TRUE
)

selenium_firefox(
  timeout = 2 * 60,
  dimensions = "1200x1200",
  headless = !interactive(),
  verbose = TRUE
)

selenium_edge(timeout = 2 * 60, dimensions = "1200x1200", verbose = TRUE)
```

```
selenium_ie(timeout = 2 * 60, dimensions = "1200x1200", verbose = TRUE)

selenium_safari(timeout = 2 * 60, dimensions = "1200x1200", verbose = TRUE)
```

Arguments

timeout	Number of seconds before selenium closes the browser
dimensions	A string in the form of "WIDTHxHEIGHT". Ex: "1800x1200"
headless	Logical which determines if the browser can run headless. Defaults to TRUE where possible.
verbose	Logical which determines if the selenium output is displayed as it's received

Details

This function assumes selenium is installed and all appropriate web browsers are installed.

Functions

- selenium_chrome(): Opens a Chrome web browser
- selenium_firefox(): Opens a Firefox web browser
- selenium_edge(): Opens an Edge web browser
- selenium_ie(): Opens an IE web browser
- selenium_safari(): Opens an Safari web browser

---

shinyjster_js *Shiny JavaScript helper*

Description

Function to be called first inside the definition of the Shiny UI.

Usage

```
shinyjster_js(..., set_timeout = TRUE)
```

Arguments

...	JavaScript text to be put in a script.
set_timeout	If TRUE (default), the JavaScript provided is executed 250 milliseconds after the document is ready. Otherwise, code is included as is.

Details

This function also includes shinyjster_ui() and wraps all JavaScript using js_script().

shinyjster_js_dependencies *shinyjster HTML Dependencies*

Description

shinyjster HTML Dependencies

Usage

```
shinyjster_js_dependencies()
```

Value

htmltools::htmlDependency's to allow shinyjster to function.

shinyjster_server *Shiny server helper*

Description

Function to be called within the shiny server

Usage

```
shinyjster_server(input, output, session = shiny::getDefaultReactiveDomain())
```

Arguments

`input`, `output`, `session`
: Shiny server function parameters

shinyjster_ui *Shiny UI helper*

Description

Function to be called first inside the definition of the Shiny UI.

Usage

```
shinyjster_ui()
```

Details

This function will add the shinyjster JS dependencies and add a text based progress bar in the bottom left corner of the application.

`testthat_shinyjster` *Test shinyjster on all browsers for shinycoreci*

Description

This method will test a single application on all of the available browsers shinyjster can test with on the given platform. For each browser, the app will be tested using `test_jster()`.

Usage

```
testthat_shinyjster(
  test_name = NULL,
  app_dir = "../../",
  ...,
  browsers = c("chrome", "firefox", "edge"),
  timeout = 2 * 60,
  dimensions = "1200x1200"
)
```

Arguments

`test_name`	suffix to add to the test name for `testthat::test_that(NAME, {})`
`app_dir`	Defaults the app in the directory above
`...`	Ignored
`browsers`	Names of each browser to be tested.
`timeout, dimensions`	Parameters to be supplied to each browser

Details

This method should be called from a test file in the `./tests/testthat/` directory.

For each browser, a new `testthat::test_that()` test will be run. This allows for browsers to not effect the other tests.

See Also

`test_jster()`

test_jster *Test shinyjster app on all browsers*

Description

This method will test your shiny application using the shinyjster code you have provided on all of the available browsers shinyjster can test with on the given platform. For each browser, each app will be tested using `run_jster_apps()`.

Usage

```
test_jster(
  apps = "../",
  browsers = c(selenium_chrome(), selenium_firefox(), if (platform() == "win" ||
   platform() == "mac") c(selenium_edge()), if (platform() == "win") c(selenium_ie()),
     if (platform() == "mac") c(selenium_safari())),
  type = c("serial", "lapply"),
  assert = TRUE,
  host = "127.0.0.1",
  port = NULL
)
```

Arguments

apps	Defaults the app in the directory above
browsers	By default, as many browsers as selenium support on the given platform
type	Single value to determine how applications are executed.
	`'serial'` Runs apps one after another using `lapply`. `port` will be random for each app unless specified.
	`'callr'` Runs apps using `callr::r_bg` using `cores` cores. `port` will be random for each app to allow concurrent execution.
	`'lapply'` Runs apps in succession using `lapply`. `port` will be random for each app unless specified.
assert	A logical value that determines if `assert_jster()` should be called on the return value
host	The IPv4 address that the application should listen on. Defaults to the `shiny.host` option, if set, or `"127.0.0.1"` if not. See Details.
port	The TCP port that the application should listen on. If the `port` is not specified, and the `shiny.port` option is set (with `options(shiny.port = XX)`), then that port will be used. Otherwise, use a random port between 3000:8000, excluding ports that are blocked by Google Chrome for being considered unsafe: 3659, 4045, 5060, 5061, 6000, 6566, 6665:6669 and 6697. Up to twenty random ports will be tried.

Value

A data frame with the columns `appDir`, `successful`, `returnValue`, and `browser`. One row of information per `browser` and `apps` combination.

See Also

`run_jster()`, `use_jster()`

`use_jster` *Create Shinyjster test file*

Description

This creates a testing file to be used with `shiny::runTests('.')`. It will call `test_jster()` which will cycle through all avaiable shinyjster selenium browsers.

Usage

```
use_jster(appDir = ".")
```

Arguments

`appDir`	Location of shiny application to test

See Also

`test_jster()`

Index