Package: brochure (via r-universe)

June 22, 2024

Title Multipage 'Shiny' Apps

Version 0.0.0.9024

Description Build multipage 'shiny' apps.

License MIT + file LICENSE

URL https://github.com/colinfay/brochure

Imports attempt (>= 0.3.1), rlang (>= 0.4.12), shiny (>= 1.6.0)

Suggests golem (>= 0.3.0), testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Repository https://thinkr-open.r-universe.dev

RemoteUrl https://github.com/ColinFay/brochure

RemoteRef HEAD

RemoteSha dde9bda06ab333855eadddc22565d86cbe0db21b

Contents

brochureApp *Create a brochureApp*

Description

This function is to be used in place of `shinyApp()`.

Usage

```
brochureApp(
  ...,
  onStart = NULL,
  options = list(),
  enableBookmarking = NULL,
  content_404 = "Not found",
  basepath = "",
  req_handlers = list(),
  res_handlers = list(),
  wrapped = shiny::tagList
)
```

Arguments

`...` a list of elements to inject in the brochureApp. **IMPORTANT NOTE** all elements which are not of class `"brochure_*"` will be injected **as is** in the page. In other word, if you use a function that return a string, the string will be added as is to the pages. The only elements that should be injected on top of `page()`s are HTML elements and/or `tagList`/`tags` that are invisible on screen (for example a `<script></script>`).

`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.

`enableBookmarking` Can be one of `"url"`, `"server"`, or `"disable"`. The default value, `NULL`, will respect the setting from any previous calls to `enableBookmarking()`. See `enableBookmarking()` for more information on bookmarking your app.

`content_404` The content to dislay when a 404 is sent

`basepath` The base path of your app. This pattern will be removed from the url, so that it matches the href of your `page()`. For example, it you have an app at `http://connect.thinkr.fr/brochure/`, and your page is names `page1`, use `basepath = "brochure"`

req_handlers	a list of functions that can manipulate the req object. These functions should take req as a parameters, and return the req object (potentially modified), or an object of class httpResponse. If any of the req_handlers return an httpResponse, this response will be sent to the browser immeditately, stopping any other code.
res_handlers	A list of functions that can manipulate the httpResponse object before it is send to the browser. Each function must take a res and req parameter.
wrapped	A UI function wrapping the Brochure UI. Default is shiny::tagList.

Value

A shiny.appobj

golem_hook *Golem Hook function*

Description

Golem Hook function

Usage

```
golem_hook(path, package_name, ...)
```

Arguments

path	Name of the folder to create the package in. This will also be used as the package name.
package_name	Package name to use. By default, golem uses basename(path). If path == '.' & package_name is not explicitly set, then basename(getwd()) will be used.
...	Arguments passed from create_golem(), unused in the default function.

Value

Used for side effect

Examples

```
if (requireNamespace("golem") & interactive()) {
  golem::create_golem("myapp", project_hook = golem_hook)
}
```

`new_page` *Add page*

Description

Module template for golem

Usage

```
new_page(name, path, export, ...)
```

Arguments

`name` The name of the module.

`path` The path to the R script where the module will be written. Note that this path will not be set by the user but via `add_module()`.

`export` Should the module be exported? Default is `FALSE`.

`...` Arguments to be passed to the `module_template` function.

Value

Used for side effect

Examples

```
if (requireNamespace("golem") & interactive()) {
  golem::add_module(name = "home", module_template = brochure::new_page)
}
```

`page` *A Brochure Page*

Description

A Brochure Page

Usage

```
page(
  href,
  ui = tagList(),
  server = function(input, output, session) {
 },
  req_handlers = list(),
  res_handlers = list()
)
```

Arguments

`href` The endpoint to serve the UI on

`ui` The UI definition of the app (for example, a call to `fluidPage()` with nested controls).

If bookmarking is enabled (see `enableBookmarking`), this must be a single argument function that returns the UI definition.

`server` A function with three parameters: `input`, `output`, and `session`. The function is called once for each session ensuring that each app is independent.

`req_handlers` a list of functions that can manipulate the `req` object. These functions should take `req` as a parameters, and return the `req` object (potentially modified), or an object of class httpResponse. If any of the req_handlers return an httpResponse, this response will be sent to the browser immeditately, stopping any other code.

`res_handlers` A list of functions that can manipulate the httpResponse object before it is send to the browser. Each function must take a `res` and `req` parameter.

Value

A list

Examples

```
library(shiny)
page(
  href = "/page2",
  ui = tagList(
    h1("This is my second page"),
    plotOutput("plotb")
  )
)
```

`parse_cookie_string` *Parse the cookie string*

Description

Parse the cookie string

Usage

```
parse_cookie_string(cookie_string)

get_cookies(session = shiny::getDefaultReactiveDomain())
```

Arguments

`cookie_string` The cookie string to parse

`session` The `{shiny}` session object.

Value

a list of cookies and values

Examples

```
parse_cookie_string("brochure_session=63422; brochure_cookie=3958")
```

`redirect` *Redirection*

Description

Redirection

Usage

```
redirect(from, to, code = 301)
```

Arguments

`from`	redirect from
`to`	redirect to
`code`	redirectin http code (one of `c(301:308, 310)`)

Value

A redirection

`server_redirect` *Do a server side redirection*

Description

Do a server side redirection

Usage

```
server_redirect(to, session = shiny::getDefaultReactiveDomain())
```

Arguments

`to`	the destination of the redirection
`session`	shiny session object, default is `shiny::getDefaultReactiveDomain()`

`set_cookie` *Middleware to set cookies*

Description

Please read https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie for more information. Description of parameters is taken from this page.

Usage

```
set_cookie(
  res,
  name,
  value,
  expires = NULL,
  max_age = NULL,
  domain = NULL,
  path = NULL,
  secure = NULL,
  http_only = NULL,
  same_site = NULL
)

remove_cookie(res, name)
```

Arguments

`res` An httpResponse object

`name` A cookie-name can be any US-ASCII characters, except control characters, spaces, or tabs. It also must not contain a separator character like the following: `( ) < > @ , ; : \ " / [ ] ? = { }`.
[ ]: R:%20

`value` A cookie-value can optionally be wrapped in double quotes and include any US-ASCII characters excluding control characters, Whitespace, double quotes, comma, semicolon, and backslash.

`expires` The maximum lifetime of the cookie as an HTTP-date timestamp. Please enter an ISO 8601 datetime format.

`max_age` Number of seconds until the cookie expires. A zero or negative number will expire the cookie immediately. If both Expires and Max-Age are set, Max-Age has precedence.

`domain` Host to which the cookie will be sent.

`path` A path that must exist in the requested URL, or the browser won't send the Cookie header.

`secure` Cookie is only sent to the server when a request is made with the https: scheme (except on localhost), and therefore is more resistent to man-in-the-middle attacks.

`http_only` Forbids JavaScript from accessing the cookie, for example, through the Document.cookie property.

`same_site` Controls whether a cookie is sent with cross-origin requests, providing some protection against cross-site request forgery attacks (CSRF).

Value

the httpResponse, with a cookie header

Examples

```
set_cookie(
  shiny:::httpResponse(),
  "this",
  12
)
```

Index