Package: mariobox (via r-universe)

June 10, 2024

Title A Framework For Packaging {plumber} APIs

Version 0.0.0.9000

Description A framework for building robust {plumber} APIs. This package contains a series of tools to build {plumber} APIs as packages.

License MIT + file LICENSE

Imports cli, fs, here, plumber, rlang, rstudioapi, usethis, utils, yaml

Suggests httr, rcmdcheck, pkgload, testthat (>= 3.0.0), withr

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

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

RemoteUrl https://github.com/ThinkR-open/mariobox

RemoteRef HEAD

RemoteSha 20abf3245d072e036daa29ccd2907e3bbc293d6d

Contents

add_endpoint *Manage endpoints*

Description

Add or remove endpoints in a mariobox project.

Usage

```
add_endpoint(name, method = "GET", open = TRUE, pkg = ".")

add_delete(name, open = TRUE, pkg = ".")

add_get(name, open = TRUE, pkg = ".")

add_patch(name, open = TRUE, pkg = ".")

add_post(name, open = TRUE, pkg = ".")

add_put(name, open = TRUE, pkg = ".")

remove_endpoint(name, method, pkg = ".")
```

Arguments

`name`	A character string. The name of the endpoint.
`method`	A character string. The name of the rest method.
`open`	A logical. Should the project be exported or not?
`pkg`	A character string. The path to package root.

Value

Nothing. Used for its side effect.

Examples

```
## Not run:
# Create a new mariobox project
path_pipo <- tempfile(pattern = "pipo")
create_mariobox(
  path = path_pipo,
  open = FALSE
)
# Add an endpoint
add_endpoint(
  name = "allo",
  method = "GET",
  open = FALSE,
  pkg = path_pipo
)
# Remove endpoint
remove_endpoint(
  name = "allo",
  pkg = path_pipo
)

## End(Not run)
```

build_plumber_file *Build the plumber.R file*

Description

Build the plumber.R file

Usage

```
build_plumber_file(pkg = ".")
```

Arguments

pkg	path to package

create_mariobox *Create a mariobox project to package a plumber API*

Description

This function will create a prepopulated package with all the necessary elements to publish a plumber API as a package.

Usage

```
create_mariobox(
  path,
  open = TRUE,
  overwrite = FALSE,
  package_name = basename(path)
)
```

Arguments

path A character string indicating the path (folder) where to create the package in. The folder name will also be used as the package name.

open A logical. Should the new project be open?

overwrite A logical. Should the already existing project be overwritten ?

package_name Package name to use. By default, mariobox uses basename(path). If path == '.' & package_name is not explicitly set, then basename(getwd()) will be used.

Value

The path to the project. Invisibly

Examples

```
path_pipo <- tempfile(pattern = "pipo")
create_mariobox(
  path = path_pipo,
  open = FALSE
)
```

http_error *Produce HTTP errors*

Description

This function is similar to stop() and is used to automatically distinguish server vs. client errors in a Plumber API.

Usage

```
http_error(status = 500L, message = NULL)
```

Arguments

status Integer, HTTP status code for the error (4xx or 5xx).

message Character, the error message (can be a vector).

Value

A condition object of class "http_error".

Examples

```
## R's default error
str(attr(try(stop("Hey, stop!")), "condition"))

## default status code 500 with default error message
str(attr(try(http_error()), "condition"))

## custom status code with default error message
str(attr(try(http_error(501L)), "condition"))

## custom status code for client error
str(attr(try(http_error(400L, "Provide valid email address.")), "condition"))
```

mario_log *Log endpoint*

Description

Log Sys.time() and endpoint name

Usage

```
mario_log(method, name)
```

Arguments

method	HTTP verb
name	Name of the endpoint

Examples

```
mario_log("GET", "health")
```

mario_try *Try an expression in a Plumber endpoint allowing error recovery*

Description

This function is similar to `try()` and is called inside a Plumber API endpoint definition. The function needs access to Plumbers response object. Use `http_error()` to automate the handling of HTTP status codes and differentiate between server vs. client errors.

Usage

```
mario_try(res, expr, silent = TRUE, ...)
```

Arguments

`res`	The HTTP response object.
`expr`	An R expression to try.
`...`	Other arguments passed to `try()`.
`message`	Logical, should the report of error messages be suppressed?

Value

The value of the expression if ’expr’ is evaluated without error, but an invisible object of class ‘"try-error"’ containing the error message, and the error condition as the ‘"condition"’ attribute, if it fails.

`new_api` *Launch the API*

Description

Launch the API

Usage

```
new_api(yaml_file)
```

Arguments

`yaml_file`	path to the yaml

Index