Package 'shiny2docker'

February 4, 2025

Title Generate Dockerfiles for 'Shiny' Applications

Version 0.0.1

Description Automates the creation of Dockerfiles for deploying 'Shiny' applications.
By integrating with 'renv' for dependency management and leveraging Docker-based
solutions, it simplifies the process of containerizing 'Shiny' apps,
ensuring reproducibility and consistency across different environments.

License MIT + file LICENSE

Imports attachment (>= 0.4.3), dockerfiler, here, yesno

Encoding UTF-8

RoxygenNote 7.3.2

URL https://github.com/VincentGuyader/shiny2docker

BugReports https://github.com/VincentGuyader/shiny2docker/issues

Suggests shiny, renv, testthat (>= 3.0.0), knitr, rmarkdown,
rstudioapi

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Vincent Guyader [aut, cre] (<https://orcid.org/0000-0003-0671-9270>)

Maintainer Vincent Guyader <vincent@thinkr.fr>

Repository CRAN

Date/Publication 2025-02-04 17:40:02 UTC

Contents

---

set_gitlab_ci   *Configure GitLab CI pipeline for Docker builds*

Description

Set the `gitlab-ci.yml` file provided by the `shiny2docker` package to the specified directory. The GitLab CI configuration is designed to build a Docker image and push the created image to the GitLab container registry.

Usage

```
set_gitlab_ci(path)
```

Arguments

path   A character string specifying the path to the directory where the `.gitlab-ci.yml` file will be copied. Defaults to the current directory ('.').

Value

A logical value indicating whether the file was successfully copied (`TRUE`) or not (`FALSE`).

Examples

```
# Copy the .gitlab-ci.yml file to the current directory
set_gitlab_ci(path=tempdir())
```

---

shiny2docker   *shiny2docker*

Description

Generate a Dockerfile for a Shiny Application

Usage

```
shiny2docker(
  path = ".",
  lockfile = file.path(path, "renv.lock"),
  output = file.path(path, "Dockerfile"),
  FROM = "rocker/geospatial",
  AS = NULL,
  sysreqs = TRUE,
  repos = c(CRAN = "https://cran.rstudio.com/"),
  expand = FALSE,
  extra_sysreqs = NULL,
```

```
    use_pak = FALSE,
    user = NULL,
    dependencies = NA,
    sysreqs_platform = "ubuntu",
    folder_to_exclude = c("renv")
)
```

Arguments

path	Character. Path to the folder containing the Shiny application (e.g., `app.R` or `ui.R` and `server.R`) along with any other necessary files.
lockfile	Character. Path to the `renv.lock` file that specifies the R package dependencies. If the `renv.lock` file does not exist, it will be created for production using the `attachment::create_renv_for_prod` function.
output	Character. Path to the generated Dockerfile. Defaults to `"Dockerfile"`.
FROM	Docker image to start FROM Default is FROM rocker/r-base
AS	The AS of the Dockerfile. Default it `NULL`.
sysreqs	boolean. If `TRUE`, the Dockerfile will contain sysreq installation.
repos	character. The URL(s) of the repositories to use for `options("repos")`.
expand	boolean. If `TRUE` each system requirement will have its own RUN line.
extra_sysreqs	character vector. Extra debian system requirements. Will be installed with apt-get install.
use_pak	boolean. If `TRUE` use pak to deal with dependencies during `renv::restore()`. FALSE by default
user	Name of the user to specify in the Dockerfile with the USER instruction. Default is `NULL`, in which case the user from the FROM image is used.
dependencies	What kinds of dependencies to install. Most commonly one of the following values:

- `NA`: only required (hard) dependencies,
- `TRUE`: required dependencies plus optional and development dependencies,
- `FALSE`: do not install any dependencies. (You might end up with a non-working package, and/or the installation might fail.)

sysreqs_platform	System requirements platform.ubuntu by default. If NULL, then the current platform is used. Can be : "ubuntu-22.04" if needed to fit with the FROM Operating System. Only debian or ubuntu based images are supported
folder_to_exclude	Folder to exclude during scan to detect packages

Details

Automate the creation of a Dockerfile tailored for deploying Shiny applications. It manages R dependencies using `renv`, generates a `.dockerignore` file to optimize the Docker build process, and leverages the `dockerfiler` package to allow further customization of the Dockerfile object before writing it to disk.

Value

An object of class dockerfiler, representing the generated Dockerfile. This object can be further manipulated using dockerfiler functions before being written to disk.

Examples

```
temp_dir <- tempfile("shiny2docker_example_")
dir.create(temp_dir)
example_app <- system.file("dummy_app", package = "shiny2docker")
file.copy(example_app, temp_dir, recursive = TRUE)

app_path <- file.path(temp_dir, "dummy_app")
if (requireNamespace("rstudioapi", quietly = TRUE) &&
rstudioapi::isAvailable()) {
  rstudioapi::filesPaneNavigate(app_path)
}

docker_obj <- shiny2docker::shiny2docker(path = app_path)

print(list.files(app_path,all.files = TRUE,no.. = TRUE))

 # Further manipulate the Dockerfile object
docker_obj$add_after(
  cmd = "ENV ENV \'MY_ENV_VAR\'=\'value\'",
  after = 3
)
docker_obj$write(file.path(app_path, "Dockerfile"))
```

Index