Package 'golem'

July 7, 2026

Title A Framework for Robust Shiny Applications

Version 1.0.1

Description An opinionated framework for building a production-ready 'Shiny' application. This package contains a series of tools for building a robust 'Shiny' application from start to finish.

License MIT + file LICENSE

URL https://thinkr-open.github.io/golem/, https://github.com/ThinkR-open/golem

BugReports https://github.com/ThinkR-open/golem/issues

Depends R (>= 3.5.0)

Imports attempt (>= 0.3.0), codetools, config, htmltools, rlang (>= 1.0.0), shiny (>= 1.5.0), utils, yaml

Suggests attachment (>= 0.3.2), callr, cli (>= 2.0.0), covr, desc, devtools, dockerfiler (>= 0.2.5), fs, httpuv, knitr, lifecycle, mockery, pkgbuild, pkgdown, pkgload (>= 1.3.0), processx, purrr, rcmdcheck, remotes, renv, rmarkdown, roxygen2, rsconnect, rstudioapi, sass, spelling, stringr, testthat (>= 3.0.0), tools, usethis (>= 1.6.0), withr

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

Config/roxygen2/version 8.0.0

RoxygenNote 8.0.0

NeedsCompilation no

Author Colin Fay [cre, aut] (ORCID: <https://orcid.org/0000-0001-7343-1846>), Vincent Guyader [aut] (ORCID: <https://orcid.org/0000-0003-0671-9270>, previous maintainer), Sébastien Rochette [aut] (ORCID: <https://orcid.org/0000-0002-1565-9313>),

Cervan Girard [aut] (ORCID: <https://orcid.org/0000-0002-4816-4624>),
Novica Nakov [ctb],
David Granjon [ctb],
Arthur Bréant [ctb],
Antoine Languillaume [ctb],
Ilya Zarubin [ctb],
ThinkR [cph]

Maintainer Colin Fay <contact@colinfay.me>

Repository CRAN

Date/Publication 2026-07-07 19:30:02 UTC

Contents

activate_js . . . 3
addins . . . 6
add_dockerfile . . . 7
add_fct . . . 11
add_github_action . . . 13
add_gitlab_ci . . . 13
add_js_file . . . 14
add_module . . . 17
add_positconnect_file . . . 18
add_resource_path . . . 19
amend_golem_config . . . 20
app_prod . . . 21
browser_button . . . 21
bundle_resources . . . 22
cat_dev . . . 23
create_golem . . . 23
detach_all_attached . . . 25
disable_autoload . . . 25
document_and_reload . . . 26
expect_shinytag . . . 27
fct_template . . . 28
fill_desc . . . 29
get_current_config . . . 30
get_golem_options . . . 31
get_golem_wd . . . 32
golem . . . 34
golem_welcome_page . . . 35
install_dev_deps . . . 35
is_golem . . . 36
is_running . . . 37
js_handler_template . . . 37
maintenance_page . . . 38
make_dev . . . 38
module_template . . . 39

pkg_name 40
project_hook 41
run_dev 41
sanity_check 42
set_golem_wd 43
use_agent_skills 44
use_claude_skills 45
use_external_js_file 46
use_favicon 49
use_module_test 50
use_readme_rmd 50
use_recommended_tests 51
use_skill 52
use_skills 53
use_utils_ui 54
with_golem_options 55

Index **56**

`activate_js` *Interact with JavaScript built-in Functions*

Description

`activate_js` is used to insert directly some JavaScript functions in your golem. By default `bundle_ressources()` load these function automatically for you.

Usage

```
activate_js()

invoke_js(fun, ..., session = shiny::getDefaultReactiveDomain())
```

Arguments

`fun`	JS function to be invoked.
`...`	JSON-like messages to be sent to the triggered JS function
`session`	The shiny session within which to call `sendCustomMessage`.

show Show an element with the jQuery selector provided. Example: `golem::invoke_js("show", "#id");`.

hide Hide an element with the jQuery selector provided. Example: `golem::invoke_js("hide", "#id")`.

showid Show an element with the id provided. Example: `golem::invoke_js("showid", "id")`.

hideid Hide an element with the id provided. Example: `golem::invoke_js("hideid", "id")`.

showclass Same as showid, but with class. Example: `golem::invoke_js("showclass", "someClass")`.

hideclass Same as hideid, but with class. Example: `golem::invoke_js("hideclass", "someClass")`.

showhref Same as showid, but with `a[href*=`. Example: `golem::invoke_js("showhref", "thinkr.fr")`.

hidehref Same as hideid, but with `a[href*=`. Example: `golem::invoke_js("hidehref", "thinkr.fr")`.

clickon Click on an element. The full jQuery selector has to be used. Example: `golem::invoke_js("clickon", "#id")`.

disable Add "disabled" to an element. The full jQuery selector has to be used. Example: `golem::invoke_js("disable", ".someClass")`.

reable Remove "disabled" from an element. The full jQuery selector has to be used. Example: `golem::invoke_js("reable", ".someClass")`.

alert Open an alert box with the message(s) provided. Example: `golem::invoke_js("alert", "WELCOME TO MY APP");`.

prompt Open a prompt box with the message(s) provided. This function takes a list with message and id `list(message = "", id = "")`. The output of the prompt will be sent to `input$id`. Example: `golem::invoke_js("prompt", list(message ="what's your name?", id = "name") )`.

confirm Open a confirm box with the message provided. This function takes a list with message and id `list(message = "", id = "")`. The output of the prompt will be sent to `input$id`. Example: `golem::invoke_js("confirm", list(message ="Are you sure you want to do that?", id = "name") )`.

Details

These JavaScript functions can be called from the server with `invoke_js`. `invoke_js` can also be used to launch any JS function created inside a Shiny JavaScript handler.

Value

Used for side-effects.

Examples

```
if (interactive()) {
  library(shiny)
  ui <- fluidPage(
    golem::activate_js(), # already loaded in your golem by `bundle_resources()`
    fluidRow(
      actionButton(inputId = "hidebutton1", label = "hide button1"),
      actionButton(inputId = "showbutton1", label = "show button1"),
      actionButton(inputId = "button1", label = "button1")
    ),
    fluidRow(
      actionButton(inputId = "hideclassA", label = "hide class A"),
      actionButton(inputId = "showclassA", label = "show class A"),
      actionButton(inputId = "buttonA1", label = "button A1", class = "A"),
```

```
    actionButton(inputId = "buttonA2", label = "button A2", class = "A"),
    actionButton(inputId = "buttonA3", label = "button A3", class = "A")
  ),
  fluidRow(
  actionButton(inputId = "clickhide", label = "click on 'hide button1' and 'hide class A'"),
  actionButton(inputId = "clickshow", label = "click on 'show button1' and 'show class A'")
  ),
  fluidRow(
    actionButton(inputId = "disableA", label = "disable class A"),
    actionButton(inputId = "reableA", label = "reable class A")
  ),
  fluidRow(
    actionButton(inputId = "alertbutton", label = "alert button"),
    actionButton(inputId = "promptbutton", label = "prompt button"),
    actionButton(inputId = "confirmbutton", label = "confirm button")
  )
)

server <- function(input, output, session) {
  observeEvent(input$hidebutton1, {
    golem::invoke_js("hideid", "button1")
  })

  observeEvent(input$showbutton1, {
    golem::invoke_js("showid", "button1")
  })

  observeEvent(input$hideclassA, {
    golem::invoke_js("hideclass", "A")
  })
  observeEvent(input$showclassA, {
    golem::invoke_js("showclass", "A")
  })

  observeEvent(input$clickhide, {
    golem::invoke_js("clickon", "#hidebutton1")
    golem::invoke_js("clickon", "#hideclassA")
  })

  observeEvent(input$clickshow, {
    golem::invoke_js("clickon", "#showbutton1")
    golem::invoke_js("clickon", "#showclassA")
  })

  observeEvent(input$disableA, {
    golem::invoke_js("disable", ".A")
  })
  observeEvent(input$reableA, {
    golem::invoke_js("reable", ".A")
  })

  observeEvent(input$alertbutton, {
    golem::invoke_js("alert", "ALERT!!")
```

```
    })

    observeEvent(input$promptbutton, {
      golem::invoke_js("prompt", list(message = "what's your name?", id = "name"))
    })
    observeEvent(input$name, {
      message(paste("input$name", input$name))
    })

    observeEvent(input$confirmbutton, {
      golem::invoke_js("confirm", list(message = "Are you sure?", id = "sure"))
    })
    observeEvent(input$sure, {
      message(paste("input$sure", input$sure))
    })
  }
  shinyApp(ui, server)
}
```

---

addins {golem} *addins*

Description

`insert_ns()` takes a selected character vector and wrap it in `ns()` The series of `go_to_*()` addins help you go to common files used in developing a {golem} application.

Usage

```
insert_ns()

go_to_start(golem_wd = golem::get_golem_wd(), wd)

go_to_dev(golem_wd = golem::get_golem_wd(), wd)

go_to_deploy(golem_wd = golem::get_golem_wd(), wd)

go_to_run_dev(golem_wd = golem::get_golem_wd(), wd)

go_to_app_ui(golem_wd = golem::get_golem_wd(), wd)

go_to_app_server(golem_wd = golem::get_golem_wd(), wd)

go_to_run_app(golem_wd = golem::get_golem_wd(), wd)
```

Arguments

golem_wd	The working directory of the {golem} application.
wd	Deprecated. Use golem_wd instead.

add_dockerfile *Create a Dockerfile for your App*

Description

Build a container containing your Shiny App. `add_dockerfile()` and `add_dockerfile_with_renv()` and `add_dockerfile_with_renv()` creates a generic Dockerfile, while `add_dockerfile_shinyproxy()`, `add_dockerfile_with_renv_shinyproxy()` , `add_dockerfile_with_renv_shinyproxy()` and `add_dockerfile_heroku()` creates platform specific Dockerfile.

Usage

```
add_dockerfile(
  path = "DESCRIPTION",
  output = "Dockerfile",
  golem_wd = get_golem_wd(),
  from = paste0("rocker/verse:", R.Version()$major, ".", R.Version()$minor),
  as = NULL,
  port = 80,
  host = "0.0.0.0",
  sysreqs = TRUE,
  repos = c(CRAN = "https://cran.rstudio.com/"),
  expand = FALSE,
  open = TRUE,
  update_tar_gz = TRUE,
  build_golem_from_source = TRUE,
  extra_sysreqs = NULL,
  pkg
)

add_dockerfile_shinyproxy(
  path = "DESCRIPTION",
  output = "Dockerfile",
  golem_wd = get_golem_wd(),
  from = paste0("rocker/verse:", R.Version()$major, ".", R.Version()$minor),
  as = NULL,
  sysreqs = TRUE,
  repos = c(CRAN = "https://cran.rstudio.com/"),
  expand = FALSE,
  open = TRUE,
  update_tar_gz = TRUE,
  build_golem_from_source = TRUE,
  extra_sysreqs = NULL,
  pkg
)

add_dockerfile_heroku(
```

```
  path = "DESCRIPTION",
  output = "Dockerfile",
  golem_wd = get_golem_wd(),
  from = paste0("rocker/verse:", R.Version()$major, ".", R.Version()$minor),
  as = NULL,
  sysreqs = TRUE,
  repos = c(CRAN = "https://cran.rstudio.com/"),
  expand = FALSE,
  open = TRUE,
  update_tar_gz = TRUE,
  build_golem_from_source = TRUE,
  extra_sysreqs = NULL,
  pkg
)

add_dockerfile_with_renv(
  golem_wd = get_golem_wd(),
  lockfile = NULL,
  output_dir = fs::path(tempdir(), "deploy"),
  distro = "focal",
  from = "rocker/verse",
  as = "builder",
  sysreqs = TRUE,
  port = 80,
  host = "0.0.0.0",
  repos = c(CRAN = "https://cran.rstudio.com/"),
  expand = FALSE,
  open = TRUE,
  document = TRUE,
  extra_sysreqs = NULL,
  update_tar_gz = TRUE,
  dockerfile_cmd = NULL,
  user = "rstudio",
  single_file = TRUE,
  set_golem.app.prod = TRUE,
  ...,
  source_folder
)

add_dockerfile_with_renv_shinyproxy(
  golem_wd = get_golem_wd(),
  lockfile = NULL,
  output_dir = fs::path(tempdir(), "deploy"),
  distro = "focal",
  from = "rocker/verse",
  as = NULL,
  sysreqs = TRUE,
  repos = c(CRAN = "https://cran.rstudio.com/"),
```

```
  expand = FALSE,
  extra_sysreqs = NULL,
  open = TRUE,
  document = TRUE,
  update_tar_gz = TRUE,
  user = "rstudio",
  single_file = TRUE,
  set_golem.app.prod = TRUE,
  ...,
  source_folder
)

add_dockerfile_with_renv_heroku(
  golem_wd = get_golem_wd(),
  lockfile = NULL,
  output_dir = fs::path(tempdir(), "deploy"),
  distro = "focal",
  from = "rocker/verse",
  as = NULL,
  sysreqs = TRUE,
  repos = c(CRAN = "https://cran.rstudio.com/"),
  expand = FALSE,
  extra_sysreqs = NULL,
  open = TRUE,
  document = TRUE,
  user = "rstudio",
  update_tar_gz = TRUE,
  single_file = TRUE,
  set_golem.app.prod = TRUE,
  ...,
  source_folder
)
```

Arguments

path	path to the DESCRIPTION file to use as an input.
output	name of the Dockerfile output.
golem_wd	path to the Package/golem source folder to deploy. default is retrieved via get_golem_wd().
from	The FROM of the Dockerfile. Default is `FROM rocker/verse` without renv.lock file passed `` `R.Version()$major`.`R.Version()$minor` `` is used as tag
as	The AS of the Dockerfile. Default it NULL.
port	The `options('shiny.port')` on which to run the App. Default is 80.

host	The options('shiny.host') on which to run the App. Default is 0.0.0.0.
sysreqs	boolean. If TRUE, RUN statements to install packages system requirements will be included in the Dockerfile.
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.
open	boolean. Should the Dockerfile/README/README be open after creation? Default is TRUE.
update_tar_gz	boolean. If TRUE and build_golem_from_source is also TRUE, an updated tar.gz is created.
build_golem_from_source	boolean. If TRUE no tar.gz is created and the Dockerfile directly mount the source folder.
extra_sysreqs	character vector. Extra debian system requirements.
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.
lockfile	path to the renv.lock file to use. default is NULL.
output_dir	folder to export everything deployment related.
distro	One of "focal", "bionic", "xenial", "centos7", or "centos8". See available distributions at https://hub.docker.com/r/rstudio/r-base/.
document	boolean. If TRUE (by default), DESCRIPTION file is updated using attachment::att_amend_desc() before creating the renv.lock file
dockerfile_cmd	What is the CMD to add to the Dockerfile. If NULL, the default, the CMD will be R -e "options('shiny.port'={port},shiny.host='{host}',golem.app.prod = {set_golem.a
user	Name of the user to specify in the Dockerfile with the USER instruction. Default is rstudio, if set to NULL no the user from the FROM image is used.
single_file	boolean. If TRUE (by default), generate a single multi-stage Dockerfile . If FALSE, produce two distinct Dockerfiles to be run sequentially for the build and production phases.
set_golem.app.prod	boolean If TRUE (by default) set options(golem.app.prod = TRUE) in dockerfile_cmd.
...	Other arguments to pass to renv::snapshot().
source_folder	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.

Value

The {dockerfiler} object, invisibly.

Note

add_dockerfile(), add_dockerfile_shinyproxy(), and add_dockerfile_heroku() are now soft deprecated; use the corresponding add_dockerfile_with_renv_*() functions instead.

Examples

```
# Add a standard Dockerfile
if (interactive() & requireNamespace("dockerfiler")) {
  add_dockerfile()
}
# Crete a 'deploy' folder containing everything needed to deploy
# the golem using docker based on {renv}
if (interactive() & requireNamespace("dockerfiler")) {
  add_dockerfile_with_renv(
    # lockfile = "renv.lock", # uncomment to use existing renv.lock file
    output_dir = "deploy"
  )
}
# Add a Dockerfile for ShinyProxy
if (interactive() & requireNamespace("dockerfiler")) {
  add_dockerfile_shinyproxy()
}

# Crete a 'deploy' folder containing everything needed to deploy
# the golem with ShinyProxy using docker based on {renv}
if (interactive() & requireNamespace("dockerfiler")) {
  add_dockerfile_with_renv(
    # lockfile = "renv.lock",# uncomment to use existing renv.lock file
    output_dir = "deploy"
  )
}

# Add a Dockerfile for Heroku
if (interactive() & requireNamespace("dockerfiler")) {
  add_dockerfile_heroku()
}
```

---

add_fct *Add fct_ and utils_ files*

Description

These functions add files in the R/ folder that starts either with `fct_` (short for function) or with `utils_`.

Usage

```
add_fct(
  name,
  module = NULL,
  golem_wd = get_golem_wd(),
  open = TRUE,
  dir_create = TRUE,
```

```
  with_test = FALSE,
  template = fct_template,
  ...,
  pkg
)

add_utils(
  name,
  module = NULL,
  golem_wd = get_golem_wd(),
  open = TRUE,
  dir_create = TRUE,
  with_test = FALSE,
  pkg
)

add_r6(
  name,
  module = NULL,
  golem_wd = get_golem_wd(),
  open = TRUE,
  dir_create = TRUE,
  with_test = FALSE,
  pkg
)
```

Arguments

name	The name of the file
module	If not NULL, the file will be module specific in the naming (you don't need to add the leading mod_).
golem_wd	Path to the root of the package. Default is get_golem_wd().
open	Should the created file be opened?
dir_create	Creates the directory if it doesn't exist, default is TRUE.
with_test	should the module be created with tests?
template	Function writing the default contents of a function file. Defaults to the built-in function template. Ignored for module-specific files.
...	Arguments passed to the template function.
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.

Value

The path to the file, invisibly.

add_github_action *Add deployment CI for GitHub Actions*

Description

Creates a minimal GitHub Actions workflow for deploying a {golem} app to Posit Connect via {rsconnect}. If needed, this function also creates a root app.R and .rscignore by calling add_positconnect_file(). The generated Posit Connect entrypoint uses {pkgload}, so {pkgload} is added to DESCRIPTION.

Usage

```
add_github_action(golem_wd = get_golem_wd(), open = TRUE)
```

Arguments

golem_wd	Path to the root of the package. Default is get_golem_wd().
open	Should the created file be opened?

Value

The path to the created workflow, invisibly.

add_gitlab_ci *Add deployment CI for GitLab*

Description

Creates a minimal GitLab CI file for deploying a {golem} app to Posit Connect via {rsconnect}. If needed, this function also creates a root app.R and .rscignore by calling add_positconnect_file(). The generated Posit Connect entrypoint uses {pkgload}, so {pkgload} is added to DESCRIPTION.

Usage

```
add_gitlab_ci(golem_wd = get_golem_wd(), open = TRUE)
```

Arguments

golem_wd	Path to the root of the package. Default is get_golem_wd().
open	Should the created file be opened?

Value

The path to the created GitLab CI file, invisibly.

add_js_file *Create Files*

Description

These functions create files inside the `inst/app` folder.

Usage

```
add_js_file(
  name,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = TRUE,
  dir_create,
  with_doc_ready = TRUE,
  template = golem::js_template,
  ...,
  pkg
)

add_js_handler(
  name,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = TRUE,
  dir_create,
  template = golem::js_handler_template,
  ...,
  pkg
)

add_js_input_binding(
  name,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = TRUE,
  dir_create,
  initialize = FALSE,
  dev = FALSE,
  events = list(name = c("change", "input"), rate_policy = c(FALSE, FALSE)),
  pkg
)

add_js_output_binding(
  name,
  golem_wd = get_golem_wd(),
```

```
  dir = "inst/app/www",
  open = TRUE,
  dir_create,
  pkg
)

add_css_file(
  name,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = TRUE,
  dir_create,
  template = golem::css_template,
  ...,
  pkg
)

add_sass_file(
  name,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = TRUE,
  dir_create,
  template = golem::sass_template,
  ...,
  pkg
)

add_empty_file(
  name,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = TRUE,
  dir_create,
  template = golem::empty_template,
  ...,
  pkg
)

add_html_template(
  name = "template.html",
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = TRUE,
  dir_create,
  pkg
)
```

```
add_partial_html_template(
  name = "partial_template.html",
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = TRUE,
  dir_create,
  pkg
)

add_ui_server_files(
  golem_wd = get_golem_wd(),
  dir = "inst/app",
  dir_create,
  pkg
)
```

Arguments

name	The name of the module.
golem_wd	Path to the root of the package. Default is get_golem_wd().
dir	Path to the dir where the file while be created.
open	Should the created file be opened?
dir_create	Deprecated. Will be removed in future versions and throws an error for now.
with_doc_ready	For JS file - Should the default file include $(document).ready()?
template	Function writing in the created file. You may overwrite this with your own template function.
...	Arguments to be passed to the template function.
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.
initialize	For JS file - Whether to add the initialize method. Default to FALSE. Some JavaScript API require to initialize components before using them.
dev	Whether to insert console.log calls in the most important methods of the binding. This is only to help building the input binding. Default is FALSE.
events	List of events to generate event listeners in the subscribe method. For instance, list(name = c("click", "keyup"), rate_policy = c(FALSE, TRUE)). The list contain names and rate policies to apply to each event. If a rate policy is found, the debounce method with a default delay of 250 ms is applied. You may edit manually according to https://shiny.posit.co/r/articles/build/building-inputs/

Value

The path to the file, invisibly.

Note

add_ui_server_files will be deprecated in future version of {golem}

See Also

js_template, js_handler_template, and css_template

---

add_module *Create a module*

Description

This function creates a module inside the R/ folder, based on a specific module structure. This function can be used outside of a {golem} project.

Usage

```
add_module(
  name,
  golem_wd = get_golem_wd(),
  open = TRUE,
  dir_create = TRUE,
  fct = NULL,
  utils = NULL,
  r6 = NULL,
  js = NULL,
  js_handler = NULL,
  export = FALSE,
  module_template = golem::module_template,
  with_test = FALSE,
  ...,
  pkg
)
```

Arguments

name	The name of the module.
golem_wd	Path to the root of the package. Default is get_golem_wd().
open	Should the created file be opened?
dir_create	Creates the directory if it doesn't exist, default is TRUE.
fct	If specified, creates a mod_fct file.
utils	If specified, creates a mod_utils file.
r6	If specified, creates a mod_class file.
js, js_handler	If specified, creates a module related JavaScript file.
export	Should the module be exported? Default is FALSE.
module_template	Function that serves as a module template.
with_test	should the module be created with tests?

...	Arguments to be passed to the module_template function.
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.

Value

The path to the file, invisibly.

Note

This function will prefix the name argument with mod_.

See Also

module_template()

---

add_positconnect_file *Add an* app.R *at the root of your package to deploy on RStudio Connect*

Description

Additionally, adds a .rscignore at the root of the {golem} project if the rsconnect package version is >= 0.8.25.

Usage

```
add_positconnect_file(golem_wd = get_golem_wd(), open = TRUE, pkg)

add_shinyappsio_file(golem_wd = get_golem_wd(), open = TRUE, pkg)

add_shinyserver_file(golem_wd = get_golem_wd(), open = TRUE, pkg)

add_rscignore_file(golem_wd = get_golem_wd(), open = TRUE, pkg)
```

Arguments

golem_wd	Path to the root of the package. Default is get_golem_wd().
open	Should the created file be opened?
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.

Value

Side-effect functions for file creation returning the path to the file, invisibly.

List of excluded files in `.rscignore`

- .here
- CODE_OF_CONDUCT.md
- LICENSE{.md}
- LICENCE{.md}
- NEWS{.md}
- README{.md,.Rmd,.HTML}
- dev
- man
- tests
- vignettes

Note

In previous versions, this function was called add_rconnect_file.

Examples

```
# Add a file for Connect
if (interactive()) {
  add_positconnect_file()
}
# Add a file for Shiny Server
if (interactive()) {
  add_shinyserver_file()
}
# Add a file for Shinyapps.io
if (interactive()) {
  add_shinyappsio_file()
}
```

`add_resource_path` *Add resource path*

Description

Add resource path

Usage

```
add_resource_path(prefix, directoryPath, warn_empty = FALSE)
```

Arguments

prefix	The URL prefix (without slashes). Valid characters are a-z, A-Z, 0-9, hyphen, period, and underscore. For example, a value of 'foo' means that any request paths that begin with '/foo' will be mapped to the given directory.
directoryPath	The directory that contains the static resources to be served.
warn_empty	Boolean. Default is FALSE. If TRUE display message if directory is empty.

Value

Used for side effects.

---

amend_golem_config *Amend golem config file*

Description

Amend golem config file

Usage

```
amend_golem_config(
  key,
  value,
  config = "default",
  golem_wd = golem::pkg_path(),
  talkative = TRUE,
  pkg
)
```

Arguments

key	key of the value to add in config
value	Name of value (NULL to read all values)
config	Name of configuration to read from. Defaults to the value of the R_CONFIG_ACTIVE environment variable ("default" if the variable does not exist).
golem_wd	Path to the root of the package. Default is get_golem_wd().
talkative	Should the messages be printed to the console?
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.

Value

Used for side effects.

app_prod *Is the app in dev mode or prod mode?*

Description

Is the app in dev mode or prod mode?

Usage

```
app_prod()

app_dev()
```

Value

TRUE or FALSE depending on the status of `getOption( "golem.app.prod")`

browser_button *Insert an hidden browser button*

Description

See https://rtask.thinkr.fr/a-little-trick-for-debugging-shiny/ for more context.

Usage

```
browser_button()
```

Value

Used for side effects. Prints the code to the console.

Note

`browser_button()` is now soft deprecated and will be removed in a future version of {golem}.

bundle_resources *Automatically serve golem external resources*

Description

This function is a wrapper around `htmltools::htmlDependency` that automatically bundles the CSS and JavaScript files in `inst/app/www` and which are created by `golem::add_css_file()` , `golem::add_js_file()` and `golem::add_js_handler()`.

Usage

```
bundle_resources(
  path,
  app_title,
  name = "golem_resources",
  version = "0.0.1",
  meta = NULL,
  head = NULL,
  attachment = NULL,
  package = NULL,
  all_files = TRUE,
  app_builder = "golem",
  with_sparkles = FALSE,
  activate_js = TRUE
)
```

Arguments

`path`	The path to the folder where the external files are located.
`app_title`	The title of the app, to be used as an application title.
`name`	Library name
`version`	Library version
`meta`	Named list of meta tags to insert into document head
`head`	Arbitrary lines of HTML to insert into the document head
`attachment`	Attachment(s) to include within the document head. See Details.
`package`	An R package name to indicate where to find the `src` directory when `src` is a relative path (see `resolveDependencies()`).
`all_files`	Whether all files under the `src` directory are dependency files. If `FALSE`, only the files specified in `script`, `stylesheet`, and `attachment` are treated as dependency files.
`app_builder`	The name of the app builder to add as a meta tag. Turn to NULL if you don't want this meta tag to be included.
`with_sparkles`	C'est quand que tu vas mettre des paillettes dans ma vie Kevin?
`activate_js`	Boolean to enable or disable the injection of JavaScript via activate_js().

Details

This function also preload `activate_js()` which allows to use preconfigured JavaScript functions via `invoke_js()`.

Value

an htmlDependency

---

cat_dev *Functions already made dev dependent*

Description

This functions will be run only if `golem::app_dev()` returns TRUE.

Usage

```
cat_dev(...)

print_dev(...)

message_dev(...)

warning_dev(...)

browser_dev(...)
```

Arguments

`...`	R objects (see 'Details' for the types of objects allowed).

Value

A modified function.

---

create_golem *Create a package for a Shiny App using* `{golem}`

Description

Create a package for a Shiny App using `{golem}`

Usage

```
create_golem(
  path,
  check_name = TRUE,
  open = TRUE,
  overwrite = FALSE,
  package_name = basename(normalizePath(path, mustWork = FALSE)),
  without_comments = FALSE,
  project_hook = golem::project_hook,
  with_git = FALSE,
  with_agents = FALSE,
  with_agents_options = NULL,
  ...
)
```

Arguments

path	Name of the folder to create the package in. This will also be used as the package name.
check_name	Should we check that the package name is correct according to CRAN requirements.
open	Boolean. Open the created project?
overwrite	Boolean. Should the already existing project be deleted and replaced?
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.
without_comments	Boolean. Start project without {golem} comments
project_hook	A function executed as a hook after project creation. Can be used to change the default {golem} structure. to override the files and content. This function is executed just after the project is created.
with_git	Boolean. Initialize git repository
with_agents	Boolean. If TRUE, the fresh package is initialized with skills files and CLAUDE/AGENTS.md
with_agents_options	named list of options passed to use_skills(); list names must match use_skills() argument names except golem_wd and interactive.
...	Arguments passed to the project_hook() function.

Value

The path, invisibly.

Note

For compatibility issue, this function turns options(shiny.autoload.r) to FALSE. See https://github.com/ThinkR-open/golem/issues/468 for more background.

detach_all_attached *Detach all attached package*

Description

Detach all attached package

Usage

```
detach_all_attached()
```

Value

TRUE, invisibly.

disable_autoload *Disabling Shiny Autoload of R Scripts*

Description

Disabling Shiny Autoload of R Scripts

Usage

```
disable_autoload(golem_wd = get_golem_wd(), pkg)
```

Arguments

golem_wd	Path to the root of the package. Default is `get_golem_wd()`.
pkg	**[Deprecated]** This argument has been replaced by `golem_wd` and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and `golem_wd` is used instead.

Value

The path to the file, invisibly.

Examples

```
if (interactive()) {
  disable_autoload()
}
```

`document_and_reload` *Document and reload your package*

Description

This function calls `rstudioapi::documentSaveAll()`, `roxygen2::roxygenise()` and `pkgload::load_all()`.

Usage

```
document_and_reload(
  golem_wd = get_golem_wd(),
  roclets = NULL,
  load_code = NULL,
  clean = FALSE,
  export_all = FALSE,
  helpers = FALSE,
  attach_testthat = FALSE,
  ...,
  pkg
)
```

Arguments

`golem_wd`	Path to the root of the package. Default is `get_golem_wd()`.
`roclets`	Character vector of roclets to use. The default, `NULL`, uses the roxygen `roclets` option, which defaults to `c("collate", "namespace", "rd")`. This will update (if needed) the `Collate` field with `update_collate()`, produce the `NAMESPACE` file with `namespace_roclet()`, and produce the Rd files with `rd_roclet()`. (Note that `update_collate()` is not technically a roclet but is still controlled with this argument for historical reasons.)
`load_code`	A function used to load all the R code in the package directory. The default, `NULL`, uses the strategy defined by the `load` roxygen option, which defaults to `load_pkgload()`. See load for more details.
`clean`	If `TRUE`, roxygen will delete all files previously created by roxygen before running each roclet.
`export_all`	If `TRUE` (the default), export all objects. If `FALSE`, export only the objects that are listed as exports in the NAMESPACE file.
`helpers`	if `TRUE` loads **testthat** test helpers.
`attach_testthat`	If `TRUE`, attach **testthat** to the search path, which more closely mimics the environment within test files.
`...`	Other arguments passed to `pkgload::load_all()`
`pkg`	**[Deprecated]** This argument has been replaced by `golem_wd` and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and `golem_wd` is used instead.

Value

Used for side-effects

`expect_shinytag` *Test helpers*

Description

These functions are designed to be used inside the tests in your Shiny app package.

Usage

```
expect_shinytag(object)

expect_shinytaglist(object)

expect_html_equal(ui, html, ...)

expect_running(sleep, R_path = NULL)
```

Arguments

`object`	the object to test
`ui`	output of an UI function
`html`	deprecated
`...`	arguments passed to `testthat::expect_snapshot()`
`sleep`	number of seconds
`R_path`	path to R. If NULL, the function will try to guess where R is.

Details

`expect_running()` only checks that an app can be **launched**: it starts a background R process that runs `run_app()` and verifies that the process is still alive after `sleep` seconds. It does **not** request any page, exercise the server logic, or detect runtime errors that happen after start-up. A passing `expect_running()` therefore means "the app starts and stays up", not "the app works".

Value

A testthat result.

`fct_template` *Golem Function Template*

Description

Function templates can be used to extend the `add_fct()` creation mechanism with your own template, so that you can be even more productive when building your `{shiny}` app. Function template functions do not aim at being called as is by users, but to be passed as an argument to the `add_fct()` function.

Usage

```
fct_template(name, path, export = FALSE, ...)
```

Arguments

`name`	The name of the generated function.
`path`	The path to the R script where the function will be written. Note that this path will not be set by the user but via `add_fct()`.
`export`	Whether the generated function should be exported.
`...`	Extra arguments, ignored by the default template.

Details

A template function can take the following arguments to be passed from `add_fct()`:

- name: the name of the function
- path: the path to the file in R/
- export: a TRUE/FALSE value

If you want your function to ignore these parameters, set `...` as the last argument of your function, then these will be ignored. See the examples section of this help.

Value

Used for side effect

See Also

`add_fct()`

Examples

```
if (interactive()) {
  my_tmpl <- function(name, path, ...) {
    # Define a template that writes to the function file
    write(name, path)
  }
  golem::add_fct(name = "custom", template = my_tmpl)
}
```

`fill_desc` *Fill your* `DESCRIPTION` *file*

Description

Generates a standard `DESCRIPTION` file as used in R packages. Also sets a series of global options inside `golem-config.yml` that will be reused inside {golem} (see `set_options` and `set_golem_options()` for details).

Usage

```
fill_desc(
  pkg_name,
  pkg_title,
  pkg_description,
  authors = person(given = NULL, family = NULL, email = NULL, role = NULL, comment =
    NULL),
  repo_url = NULL,
  pkg_version = "0.0.0.9000",
  pkg = get_golem_wd(),
  author_first_name = NULL,
  author_last_name = NULL,
  author_email = NULL,
  author_orcid = NULL,
  set_options = TRUE
)
```

Arguments

`pkg_name`	The name of the package
`pkg_title`	The title of the package
`pkg_description`	Description of the package
`authors`	a character string (or vector) of class person (see `person()` for details)
`repo_url`	URL (if needed)
`pkg_version`	The version of the package. Default is 0.0.0.9000

pkg Path to look for the DESCRIPTION. Default is `get_golem_wd()`.

author_first_name
Deprecated: use `authors = person(given = ...)` instead

author_last_name
Deprecated: use `authors = person(family = ...)` instead

author_email Deprecated: use `authors = person(email = ...)` instead

author_orcid Deprecated: use `authors = person(comment = c(ORCID = ...))` instead

set_options logical; the default `TRUE` sets all recommended options but this can be suppressed with `FALSE`. For details on the exact behaviour see the help `set_golem_options()`.

Value

The {desc} object, invisibly.

---

get_current_config *Return path to the {golem} config-file*

Description

This function tries to find the current config file, being either inst/golem-config.yml or the GOLEM_CONFIG_PATH env var

Usage

```
get_current_config(path = getwd())
```

Arguments

path character string giving the path to start looking for the config; the usual value is the {golem}-package top-level directory but a user supplied config is now supported (see **Details** for how to use this feature).

Details

In most cases this function simply returns the path to the default golem-config file located under "inst/golem-config.yml". That config comes in `yml`-format, see the Engineering Production-Grade Shiny Apps for further details on its format and how to set options therein.

Advanced app developers may benefit from having an additional user config-file. This is achieved with setting the GOLEM_CONFIG_PATH env var.

Value

character string giving the path to the {golem} config-file

get_golem_options *Get all or one golem options*

Description

This function is to be used inside the server and UI from your app, in order to call the parameters passed to `run_app()`.

Usage

```
get_golem_options(which = NULL)
```

Arguments

`which`	NULL (default), or the name of an option

Value

The value of the option.

Examples

```
# Define and use golem_options
if (interactive()) {
  # 1. Pass parameters directly to `run_app`

  run_app(
    title = "My Golem App",
    content = "something"
  )

  # 2. Get the values
  # 2.1 from the UI side

  h1(get_golem_options("title"))

  # 2.2 from the server-side

  output$param <- renderPrint({
    paste("param content = ", get_golem_options("content"))
  })

  output$param_full <- renderPrint({
    get_golem_options() # list of all golem options as a list.
  })

  # 3. If needed, to set default value, edit `run_app` like this :

  run_app <- function(
  title = "this",
```

```
  content = "that",
  ...
  ) {
    with_golem_options(
      app = shinyApp(
        ui = app_ui,
        server = app_server
      ),
      golem_opts = list(
        title = title,
        content = content,
        ...
      )
    )
  }
}
```

---

get_golem_wd {golem} *options*

Description

Set and get a series of options to be used with {golem}. These options are found inside the golem-config.yml file, found in most cases inside the inst folder.

Usage

```
get_golem_wd(use_parent = TRUE, golem_wd = golem::pkg_path(), pkg)

get_golem_name(
 config = Sys.getenv("GOLEM_CONFIG_ACTIVE", Sys.getenv("R_CONFIG_ACTIVE", "default")),
  use_parent = TRUE,
  golem_wd = golem::pkg_path(),
  pkg
)

get_golem_version(
 config = Sys.getenv("GOLEM_CONFIG_ACTIVE", Sys.getenv("R_CONFIG_ACTIVE", "default")),
  use_parent = TRUE,
  golem_wd = golem::pkg_path(),
  pkg
)

set_golem_name(
  name = golem::pkg_name(),
  golem_wd = golem::pkg_path(),
  talkative = TRUE,
```

```
  old_name = golem::pkg_name(),
  pkg
)

set_golem_version(
  version = golem::pkg_version(),
  golem_wd = golem::pkg_path(),
  talkative = TRUE,
  pkg
)

set_golem_options(
  golem_name = golem::pkg_name(),
  golem_version = golem::pkg_version(),
  golem_wd = golem::pkg_path(),
  app_prod = FALSE,
  talkative = TRUE,
  config_file = golem::get_current_config(golem_wd)
)
```

Arguments

use_parent	TRUE to scan parent directories for configuration files if the specified config file isn't found.
golem_wd	Working directory of the current golem package.
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.
config	Name of configuration to read from. Defaults to the value of the R_CONFIG_ACTIVE environment variable ("default" if the variable does not exist).
name	The name of the app
talkative	Should the messages be printed to the console?
old_name	The old name of the app, used when changing the name
version	The version of the app
golem_name	Name of the current golem.
golem_version	Version of the current golem.
app_prod	Is the {golem} in prod mode?
config_file	path to the {golem} config file

Value

Used for side-effects for the setters, and values from the config in the getters.

Set Functions

- `set_golem_options()` sets all the options, with the defaults from the functions below.
- `set_golem_wd()` defaults to `golem::golem_wd()`, which is the package root when starting a golem.
- `set_golem_name()` defaults `golem::pkg_name()`
- `set_golem_version()` defaults `golem::pkg_version()`

Get Functions

Reads the information from `golem-config.yml`

- `get_golem_wd()`
- `get_golem_name()`
- `get_golem_version()`

`golem` *A package for building Shiny App*

Description

Read more about building big shiny apps at https://engineering-shiny.org/.

Author(s)

Maintainer: Colin Fay <contact@colinfay.me> (ORCID)

Authors:

- Colin Fay <contact@colinfay.me> (ORCID)
- Vincent Guyader <vincent@thinkr.fr> (ORCID) (previous maintainer)
- Sébastien Rochette <sebastien@thinkr.fr> (ORCID)
- Cervan Girard <cervan@thinkr.fr> (ORCID)

Other contributors:

- Novica Nakov <nnovica@gmail.com> [contributor]
- David Granjon <dgranjon@ymail.com> [contributor]
- Arthur Bréant <arthur@thinkr.fr> [contributor]
- Antoine Languillaume <antoine@thinkr.fr> [contributor]
- Ilya Zarubin <zarubin@wiso.uni-koeln.de> [contributor]
- ThinkR [copyright holder]

See Also

Useful links:

- https://thinkr-open.github.io/golem/
- https://github.com/ThinkR-open/golem
- Report bugs at https://github.com/ThinkR-open/golem/issues

`golem_welcome_page` *Welcome Page*

Description

Welcome Page

Usage

```
golem_welcome_page()
```

Value

A welcome page for your {golem} app

`install_dev_deps` *Install* {golem} *dev dependencies*

Description

This function will run rlang::check_installed() on:

- usethis
- pkgload
- dockerfiler
- devtools
- roxygen2
- attachment
- rstudioapi
- fs
- desc
- pkgbuild
- processx
- rsconnect
- testthat

Usage

```
install_dev_deps(dev_deps, force_install = FALSE, ...)
```

Arguments

dev_deps	optional character vector of packages to install
force_install	If force_install is TRUE, then the user is not interactively asked to install them.
...	further arguments passed to the install function.

Value

Used for side-effects

Examples

```
if (interactive()) {
  install_dev_deps()
}
```

---

is_golem *Is the directory a golem-based app?*

Description

Trying to guess if path is a golem-based app.

Usage

```
is_golem(path = getwd())
```

Arguments

path	Path to the directory to check. Defaults to the current working directory.

Value

A boolean, TRUE if the directory is a golem-based app, FALSE else.

Examples

```
is_golem()
```

is_running *Is the running app a golem app?*

Description

Note that this will return TRUE only if the application has been launched with with_golem_options()

Usage

```
is_running()
```

Value

TRUE if the running app is a {golem} based app, FALSE otherwise.

Examples

```
is_running()
```

js_handler_template *Golem's default custom templates*

Description

These functions do not aim at being called as is by users, but to be passed as an argument to the add_js_handler() function.

Usage

```
js_handler_template(path, name = "fun", code = " ")

js_template(path, code = " ")

css_template(path, code = " ")

sass_template(path, code = " ")

empty_template(path, code = " ")
```

Arguments

path	The path to the JS script where this template will be written.
name	Shiny's custom handler name.
code	JavaScript code to be written in the function.

Value

Used for side effect

See Also

`add_js_handler()`

`maintenance_page` *maintenance_page*

Description

A default html page for maintenance mode

Usage

```
maintenance_page()
```

Details

see the vignette `vignette("f_extending_golem", package = "golem")` for details.

Value

an html_document

`make_dev` *Make a function dependent to dev mode*

Description

The function returned will be run only if `golem::app_dev()` returns TRUE.

Usage

```
make_dev(fun)
```

Arguments

`fun`	A function

Value

Used for side-effects

`module_template` *Golem Module Template Function*

Description

Module template can be used to extend golem module creation mechanism with your own template, so that you can be even more productive when building your `{shiny}` app. Module template functions do not aim at being called as is by users, but to be passed as an argument to the `add_module()` function.

Usage

```
module_template(name, path, export, ph_ui = " ", ph_server = " ", ...)
```

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`.
`ph_ui, ph_server`	Texts to insert inside the modules UI and server. For advanced use.
`...`	Arguments to be passed to the `module_template` function.

Details

Module template functions are a way to define your own template function for module. A template function that can take the following arguments to be passed from `add_module()`:

- name: the name of the module
- path: the path to the file in R/
- export: a TRUE/FALSE set by the `export` param of `add_module()`

If you want your function to ignore these parameters, set `...` as the last argument of your function, then these will be ignored. See the examples section of this help.

Value

Used for side effect

See Also

`add_module()`

Examples

```
if (interactive()) {
  my_tmpl <- function(name, path, ...) {
    # Define a template that write to the
    # module file
    write(name, path)
  }
  golem::add_module(name = "custom", module_template = my_tmpl)

  my_other_tmpl <- function(name, path, ...) {
    # Copy and paste a file from somewhere
    file.copy(..., path)
  }
  golem::add_module(name = "custom", module_template = my_other_tmpl)
}
```

---

pkg_name *Package tools*

Description

These are functions to help you navigate inside your project while developing

Usage

```
pkg_name(golem_wd = get_golem_wd(), path)

pkg_version(golem_wd = get_golem_wd(), path)

pkg_path(golem_wd = getwd())
```

Arguments

golem_wd	Path to use to read the DESCRIPTION
path	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.

Value

The value of the entry in the DESCRIPTION file

project_hook *Project Hook*

Description

Project hooks allow to define a function run just after {golem} project creation.

Usage

```
project_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 effects

Examples

```
if (interactive()) {
  my_proj <- function(...) {
    unlink("dev/", TRUE, TRUE)
  }
  create_golem("ici", project_template = my_proj)
}
```

run_dev *Run the* dev/run_dev.R *file*

Description

The default file="dev/run_dev.R" launches your {golem} app with a bunch of useful options. The file content can be customized and file-name and path changed as long as the argument combination of file and pkg are supplied correctly: the file-path is a relative path to a {golem}-package root pkg. An error is thrown if pkg/file cannot be found.

Usage

```
run_dev(
  file = "dev/run_dev.R",
  golem_wd = get_golem_wd(),
  save_all = TRUE,
  install_required_packages = TRUE,
  pkg
)
```

Arguments

file	String with (relative) file path to a run_dev.R-file
golem_wd	Path to the root of the package. Default is get_golem_wd().
save_all	Boolean; if TRUE saves all open files before sourcing file
install_required_packages	Boolean; if TRUE install the packages used in run_dev.R-file
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.

Details

The function run_dev() is typically used to launch a shiny app by sourcing the content of an appropriate run_dev-file. Carefully read the content of dev/run_dev.R when creating your custom run_dev-file. It has already many useful settings including a switch between production/development, reloading the package in a clean R environment before running the app etc.

Value

pure side-effect function; returns invisibly

---

sanity_check *Sanity check for R files in the project*

Description

This function is used check for any 'browser()" or commented #TODO / #TOFIX / #BUG in the code

Usage

```
sanity_check(golem_wd = get_golem_wd(), pkg)
```

Arguments

golem_wd	Path to the root of the package. Default is get_golem_wd().
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.

Value

A DataFrame if any of the words has been found.

---

set_golem_wd *Set the golem working directory*

Description

Set the golem_wd value (the working directory of the current golem package) inside the golem-config.yml file.

Usage

```
set_golem_wd(
  new_golem_wd = golem::pkg_path(),
  current_golem_wd = golem::pkg_path(),
  talkative = TRUE,
  golem_wd,
  pkg
)
```

Arguments

new_golem_wd, current_golem_wd	New & current directory, to be used in set_golem_wd().
talkative	Should the messages be printed to the console?
golem_wd	**[Deprecated]** This argument has been replaced by new_golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and new_golem_wd is used instead.
pkg	**[Deprecated]** This argument has been replaced by current_golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and current_golem_wd is used instead.

Value

Used for side-effects, and returns the golem_wd path invisibly.

use_agent_skills *Add Agent Skills to a golem Project*

Description

Add Agent Skills to a golem Project

Usage

```
use_agent_skills(
  source = c("ask", "local", "remote"),
  skills = NULL,
  main_md_files = c("ask", "yes", "no"),
  overwrite = c("ask", "overwrite", "skip", "abort"),
  golem_wd = get_golem_wd(),
  interactive = rlang_is_interactive()
)
```

Arguments

source	Where to install agent skills from. Use "ask" for the interactive menu.
skills	Skills to install. Use NULL for the interactive selector, or "all" to install all available skills.
main_md_files	Whether to also add AGENTS.md / CLAUDE.md. Use "ask" for the interactive menu.
overwrite	How to handle existing files. Use "ask" for the interactive menu.
golem_wd	Path to the golem project where files should be copied.
interactive	Whether "ask" values and skills = NULL should prompt. In non-interactive mode, those values use local defaults.

Value

A list of selected options and copied paths, invisibly.

See Also

use_skills(), use_agent_implement(), use_claude_skills(), use_skill()

`use_claude_skills` *Add Claude Skills to a golem Project*

Description

Add Claude Skills to a golem Project

Usage

```
use_claude_skills(
  source = c("ask", "local", "remote"),
  skills = NULL,
  main_md_files = c("ask", "yes", "no"),
  overwrite = c("ask", "overwrite", "skip", "abort"),
  golem_wd = get_golem_wd(),
  interactive = rlang_is_interactive()
)
```

Arguments

`source`	Where to install agent skills from. Use `"ask"` for the interactive menu.
`skills`	Skills to install. Use `NULL` for the interactive selector, or `"all"` to install all available skills.
`main_md_files`	Whether to also add `AGENTS.md` / `CLAUDE.md`. Use `"ask"` for the interactive menu.
`overwrite`	How to handle existing files. Use `"ask"` for the interactive menu.
`golem_wd`	Path to the golem project where files should be copied.
`interactive`	Whether `"ask"` values and `skills = NULL` should prompt. In non-interactive mode, those values use local defaults.

Value

A list of selected options and copied paths, invisibly.

See Also

`use_skills()`, `use_agent_implement()`, `use_agent_skills()`, `use_skill()`

use_external_js_file *Use Files*

Description

These functions download files from external sources and put them inside the `inst/app/www` directory. The `use_internal_` functions will copy internal files, while `use_external_` will try to download them from a remote location.

Usage

```
use_external_js_file(
  url,
  name = NULL,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = FALSE,
  dir_create,
  pkg,
  replace = FALSE
)

use_external_css_file(
  url,
  name = NULL,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = FALSE,
  dir_create,
  pkg,
  replace = FALSE
)

use_external_html_template(
  url,
  name = "template.html",
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = FALSE,
  dir_create,
  extract = c("ask", "yes", "no"),
  delete_zip = c("ask", "yes", "no"),
  replace = FALSE
)

use_external_file(
  url,
```

```
  name = NULL,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = FALSE,
  dir_create,
  pkg,
  replace = FALSE
)

use_bundled_html(
  url,
  name = NULL,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = FALSE,
  extract = c("ask", "yes", "no"),
  delete_zip = c("ask", "yes", "no"),
  replace = FALSE
)

use_internal_js_file(
  path,
  name = NULL,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = FALSE,
  dir_create,
  pkg
)

use_internal_css_file(
  path,
  name = NULL,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = FALSE,
  dir_create,
  pkg
)

use_internal_html_template(
  path,
  name = "template.html",
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = FALSE,
  dir_create,
  pkg
```

```
)

use_internal_file(
  path,
  name = NULL,
  golem_wd = get_golem_wd(),
  dir = "inst/app/www",
  open = FALSE,
  dir_create,
  pkg
)
```

Arguments

url	String representation of URL for the file to be downloaded
name	The name of the module.
golem_wd	Path to the root of the package. Default is get_golem_wd().
dir	Path to the dir where the file while be created.
open	Should the created file be opened?
dir_create	Creates the directory if it doesn't exist, default is TRUE.
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.
replace	Boolean. If TRUE, an existing file (or, for use_bundled_html(), an existing bundle directory) at the target location is overwritten. Defaults to FALSE, in which case the function aborts if the target already exists.
extract	Whether to extract a downloaded HTML zip bundle. Use "ask" to prompt. Only used by use_bundled_html() and by use_external_html_template() when url points to a .zip archive.
delete_zip	Whether to delete the raw HTML zip after extraction. Use "ask" to prompt. Only used by use_bundled_html() and by use_external_html_template() when url points to a .zip archive.
path	String representation of the local path for the file to be implemented (use_file only)

Value

The path to the file, invisibly.

Note

See ?htmltools::htmlTemplate and https://shiny.posit.co/r/articles/build/templates/ for more information about htmlTemplate.

use_favicon *Add a favicon to your shinyapp*

Description

This function adds the favicon from `ico` to your shiny app.

Usage

```
use_favicon(path, golem_wd = get_golem_wd(), method = "curl", pkg)

remove_favicon(path = "inst/app/www/favicon.ico")

favicon(
  ico = "favicon",
  rel = "shortcut icon",
  resources_path = "www",
  ext = "ico"
)
```

Arguments

`path`	Path to your favicon file (.ico or .png)
`golem_wd`	Path to the root of the package. Default is `get_golem_wd()`.
`method`	Method to be used for downloading files, 'curl' is default see `utils::download.file()`.
`pkg`	**[Deprecated]** This argument has been replaced by `golem_wd` and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and `golem_wd` is used instead.
`ico`	path to favicon file
`rel`	rel
`resources_path`	prefix of the resource path of the app
`ext`	the extension of the favicon

Value

Used for side-effects.

An HTML tag.

Examples

```
if (interactive()) {
  use_favicon()
  use_favicon(path = "path/to/your/favicon.ico")
}
```

use_module_test *Add a test file for a module*

Description

Add a test file for in module, with the new testServer structure.

Usage

```
use_module_test(name, golem_wd = get_golem_wd(), open = TRUE, pkg)
```

Arguments

name	The name of the module.
golem_wd	Path to the root of the package. Default is `get_golem_wd()`.
open	Should the created file be opened?
pkg	**[Deprecated]** This argument has been replaced by `golem_wd` and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and `golem_wd` is used instead.

Value

Used for side effect. Returns the path invisibly.

use_readme_rmd *Generate a README.Rmd*

Description

Generate a README.Rmd

Usage

```
use_readme_rmd(
  open = rlang::is_interactive(),
  pkg_name = golem::get_golem_name(),
  overwrite = FALSE,
  golem_wd = golem::get_golem_wd(),
  pkg
)
```

Arguments

open	Open the newly created file for editing? Happens in RStudio, if applicable, or via utils::file.edit() otherwise.
pkg_name	The name of the package
overwrite	an optional logical flag; if TRUE, overwrite existing README.Rmd, else throws an error if README.Rmd exists
golem_wd	Path to the root of the package. Default is get_golem_wd().
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.

Value

pure side-effect function that generates template README.Rmd

---

use_recommended_tests *Add recommended elements*

Description

Add recommended elements

Usage

```
use_recommended_tests(
  golem_wd = get_golem_wd(),
  spellcheck = TRUE,
  vignettes = TRUE,
  lang = "en-US",
  error = FALSE,
  pkg
)
```

Arguments

golem_wd	Path to the root of the package. Default is get_golem_wd().
spellcheck	Whether or not to use a spellcheck test.
vignettes	Logical, TRUE to spell check all rmd and rnw files in the vignettes/ folder.
lang	Preferred spelling language. Usually either "en-US" or "en-GB".
error	Logical, indicating whether the unit test should fail if spelling errors are found. Defaults to FALSE, which does not error, but prints potential spelling errors
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.

Value

Used for side-effects.

`use_skill` *Add a Single Skill to Installed Agent Specifications*

Description

Add a Single Skill to Installed Agent Specifications

Usage

```
use_skill(
  name,
  source = NULL,
  overwrite = c("ask", "overwrite", "skip", "abort"),
  golem_wd = get_golem_wd(),
  interactive = rlang_is_interactive()
)
```

Arguments

`name`	Skill name to install.
`source`	Where to install agent skills from. If `NULL`, prompt interactively; falls back to `"local"` in non-interactive mode.
`overwrite`	How to handle existing files. Use `"ask"` for the interactive menu.
`golem_wd`	Path to the golem project where files should be copied.
`interactive`	Whether `"ask"` values and `source = NULL` should prompt. In non-interactive mode, those values use local defaults.

Value

A list of selected options and copied paths, invisibly.

See Also

`use_skills()`, `use_agent_implement()`, `use_agent_skills()`, `use_claude_skills()`

use_skills *Add Skills to a golem Project*

Description

Add Skills to a golem Project

Usage

```
use_skills(
  source = c("ask", "local", "remote"),
  agent_specs = c("ask", "claude", "agents", "both"),
  skills = NULL,
  main_md_files = c("ask", "yes", "no"),
  overwrite = c("ask", "overwrite", "skip", "abort"),
  golem_wd = get_golem_wd(),
  interactive = rlang_is_interactive()
)
```

Arguments

source	Where to install agent skills from. Use "ask" for the interactive menu.
agent_specs	Which agent specifications to install. Use "ask" for the interactive menu.
skills	Skills to install. Use NULL for the interactive selector, or "all" to install all available skills.
main_md_files	Whether to also add AGENTS.md / CLAUDE.md. Use "ask" for the interactive menu.
overwrite	How to handle existing files. Use "ask" for the interactive menu.
golem_wd	Path to the golem project where files should be copied.
interactive	Whether "ask" values and skills = NULL should prompt. In non-interactive mode, those values use local defaults.

Value

A list of selected options and copied paths, invisibly.

Variants

- use_skills() is the main entry point — pick the agent target via agent_specs ("claude", "agents", or "both").
- use_agent_skills() and use_claude_skills() are convenience wrappers for use_skills(agent_specs = "agents") and use_skills(agent_specs = "claude").
- use_skill() adds a single skill to an already-installed agent target, without touching CLAUDE.md / AGENTS.md.

See Also

use_agent_implement(), use_agent_skills(), use_claude_skills(), use_skill()

---

use_utils_ui *Use the utils files*

Description

use_utils_ui Copies the golem_utils_ui.R to the R folder.

use_utils_server Copies the golem_utils_server.R to the R folder.

Usage

```
use_utils_ui(golem_wd = get_golem_wd(), with_test = FALSE, pkg)

use_utils_test_ui(golem_wd = get_golem_wd(), pkg)

use_utils_server(golem_wd = get_golem_wd(), with_test = FALSE, pkg)

use_utils_test_ui(golem_wd = get_golem_wd(), pkg)

use_utils_test_server(golem_wd = get_golem_wd(), pkg)
```

Arguments

golem_wd	Path to the root of the package. Default is get_golem_wd().
with_test	should the module be created with tests?
pkg	**[Deprecated]** This argument has been replaced by golem_wd and is kept here for backward compatibility. Providing a value to this argument has no effect: the value is silently ignored, and golem_wd is used instead.

Value

Used for side-effects.

`with_golem_options` *Add Golem options to a Shiny App*

Description

You'll probably never have to write this function as it is included in the golem template created on launch.

Usage

```
with_golem_options(
  app,
  golem_opts,
  maintenance_page = golem::maintenance_page,
  print = FALSE
)
```

Arguments

`app`	the app object.
`golem_opts`	A list of options to be added to the app
`maintenance_page`	an html_document or a shiny tag list. Default is golem template.
`print`	Whether or not to print the app. Default is to `FALSE`, which should be what you need 99.99% of the time. In case you need to actively print() the app object, you can set it to `TRUE`.

Value

a shiny.appObj object

Index

activate_js, 3
activate_js(), *23*
add_css_file *(*add_js_file*)*, 14
add_dockerfile, 7
add_dockerfile_heroku *(*add_dockerfile*)*, 7
add_dockerfile_shinyproxy *(*add_dockerfile*)*, 7
add_dockerfile_with_renv *(*add_dockerfile*)*, 7
add_dockerfile_with_renv_heroku *(*add_dockerfile*)*, 7
add_dockerfile_with_renv_shinyproxy *(*add_dockerfile*)*, 7
add_empty_file *(*add_js_file*)*, 14
add_fct, 11
add_fct(), *28*
add_github_action, 13
add_gitlab_ci, 13
add_html_template *(*add_js_file*)*, 14
add_js_file, 14
add_js_handler *(*add_js_file*)*, 14
add_js_handler(), *38*
add_js_input_binding *(*add_js_file*)*, 14
add_js_output_binding *(*add_js_file*)*, 14
add_module, 17
add_module(), *39*
add_partial_html_template *(*add_js_file*)*, 14
add_positconnect_file, 18
add_positconnect_file(), *13*
add_r6 *(*add_fct*)*, 11
add_rconnect_file *(*add_positconnect_file*)*, 18
add_resource_path, 19
add_rscignore_file *(*add_positconnect_file*)*, 18
add_sass_file *(*add_js_file*)*, 14
add_shinyappsio_file *(*add_positconnect_file*)*, 18
add_shinyserver_file *(*add_positconnect_file*)*, 18
add_ui_server_files *(*add_js_file*)*, 14
add_utils *(*add_fct*)*, 11
addins, 6
amend_golem_config, 20
app_dev *(*app_prod*)*, 21
app_prod, 21
attachment::att_amend_desc(), *10*

browser_button, 21
browser_dev *(*cat_dev*)*, 23
bundle_resources, 22

cat_dev, 23
create_golem, 23
css_template, *17*
css_template *(*js_handler_template*)*, 37

detach_all_attached, 25
disable_autoload, 25
document_and_reload, 26

empty_template *(*js_handler_template*)*, 37
expect_html_equal *(*expect_shinytag*)*, 27
expect_running *(*expect_shinytag*)*, 27
expect_shinytag, 27
expect_shinytaglist *(*expect_shinytag*)*, 27

favicon *(*use_favicon*)*, 49
fct_template, 28
fill_desc, 29

get_current_config, 30
get_golem_name *(*get_golem_wd*)*, 32
get_golem_options, 31
get_golem_version *(*get_golem_wd*)*, 32
get_golem_wd, 32
get_golem_wd(), *9*

go_to_app_server (addins), 6
go_to_app_ui (addins), 6
go_to_deploy (addins), 6
go_to_dev (addins), 6
go_to_run_app (addins), 6
go_to_run_dev (addins), 6
go_to_start (addins), 6
golem, 34
golem-package (golem), 34
golem_welcome_page, 35

insert_ns (addins), 6
install_dev_deps, 35
invoke_js (activate_js), 3
invoke_js(), *23*
is_golem, 36
is_running, 37

js_handler_template, *17*, 37
js_template, *17*
js_template (js_handler_template), 37

load, *26*
load_pkgload(), *26*

maintenance_page, 38
make_dev, 38
message_dev (cat_dev), 23
module_template, 39
module_template(), *18*

namespace_roclet(), *26*

person(), *29*
pkg_name, 40
pkg_path (pkg_name), 40
pkg_version (pkg_name), 40
print_dev (cat_dev), 23
project_hook, 41

rd_roclet(), *26*
remove_favicon (use_favicon), 49
renv::snapshot(), *10*
resolveDependencies(), *22*
roclets, *26*
run_dev, 41

sanity_check, 42
sass_template (js_handler_template), 37
set_golem_name (get_golem_wd), 32
set_golem_options (get_golem_wd), 32
set_golem_options(), *29, 30*
set_golem_version (get_golem_wd), 32
set_golem_wd, 43

update_collate(), *26*
use_agent_implement(), *44, 45*, *52*, *54*
use_agent_skills, 44
use_agent_skills(), *45*, *52*, *54*
use_bundled_html
        (use_external_js_file), 46
use_claude_skills, 45
use_claude_skills(), *44*, *52*, *54*
use_external_css_file
        (use_external_js_file), 46
use_external_file
        (use_external_js_file), 46
use_external_html_template
        (use_external_js_file), 46
use_external_js_file, 46
use_favicon, 49
use_internal_css_file
        (use_external_js_file), 46
use_internal_file
        (use_external_js_file), 46
use_internal_html_template
        (use_external_js_file), 46
use_internal_js_file
        (use_external_js_file), 46
use_module_test, 50
use_readme_rmd, 50
use_recommended_tests, 51
use_skill, 52
use_skill(), *44, 45*, *54*
use_skills, 53
use_skills(), *24*, *44, 45*, *52*
use_utils_server (use_utils_ui), 54
use_utils_test_server (use_utils_ui), 54
use_utils_test_ui (use_utils_ui), 54
use_utils_ui, 54
utils::download.file(), *49*
utils::file.edit(), *51*

warning_dev (cat_dev), 23
with_golem_options, 55