Package ‘golem’

March 5, 2020

Title A Framework for Robust Shiny Applications

Version 0.2.1

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://github.com/ThinkR-open/golem

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

Depends R (>= 3.0)

Imports attempt (>= 0.3.0), cli, config, crayon, desc, dockerfiler, fs, here, htmltools, jsonlite, pkgload, remotes, rlang, roxygen2, rstudioapi, shiny, testthat, usethis, utils, yaml

Suggests covr, devtools, glue, knitr, pkgbuild, pkgdown, purrr, remdcheck, rmarkdown, rsconnect, spelling, stringr, withr

VignetteBuilder knitr

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.0.2

NeedsCompilation no

Author Vincent Guyader [cre, aut] (<https://orcid.org/0000-0003-0671-9270>), Colin Fay [aut] (<https://orcid.org/0000-0001-7343-1846>), Sébastien Rochette [aut] (<https://orcid.org/0000-0002-1565-9313>), Cervan Girard [aut] (<https://orcid.org/0000-0002-4816-4624>), Novica Nakov [ctb], ThinkR [cph]

Maintainer Vincent Guyader <vincent@thinkr.fr>

Repository CRAN

Date/Publication 2020-03-05 15:00:02 UTC
activate_js

Interact with JavaScript built-in Functions

activate_js is used in your UI to insert directly the JavaScript functions contained in golem. These 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.

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.
- **hide**: Hide an element with the jQuery selector provided.
- **showid**: Show an element with the id provided.
- **hideid**: Hide an element with the id provided.
- **showclass**: Same as `showid`, but with class.
- **hideclass**: Same as `hideid`, but with class.
- **showhref**: Same as `showid`, but with `a[href*=`.
- **hidehref**: Same as `hideid`, but with `a[href*=`.
- **clickon**: Click on an element. The full jQuery selector has to be used.
- **disable**: Add "disabled" to an element. The full jQuery selector has to be used.
- **reable**: Remove "disabled" from an element. The full jQuery selector has to be used.
- **alert**: Open an alert box with the message(s) provided.
- **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$id`.
- **confirm**: Open a confirm box with the message provided. This function takes a list with message and id `list(message = "", id = "")`. The output of the prompt will be sent to `input$id`.

---

**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()`
- `go_to_dev()`
- `go_to_deploy()`
- `go_to_run_dev()`
add_dockerfile

Description
Build a container containing your Shiny App. add_dockerfile() creates a "classical" Dockerfile, while add_dockerfile_shinyproxy() and add_dockerfile_heroku() creates platform specific Dockerfile.

Usage

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

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

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>path to the DESCRIPTION file to use as an input.</td>
</tr>
<tr>
<td>output</td>
<td>name of the Dockerfile output.</td>
</tr>
<tr>
<td>pkg</td>
<td>Path to the root of the package. Default is <code>get_golem_wd()</code>.</td>
</tr>
</tbody>
</table>
as
port
host
sysreqs
repos
expand
open
update_tar.gz
build_golem_from_source

Examples

# Add a standard Dockerfile
if (interactive()){
  add_dockerfile()
}

# Add a Dockerfile for ShinyProxy
if (interactive()){
  add_dockerfile_shinyproxy()
}

# Add a Dockerfile for Heroku
if (interactive()){
  add_dockerfile_heroku()
}

Description

These function adds files in the R/ folder that starts either with fct_ or with utils_.

Usage

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

add_utils(name, module = NULL, pkg = get_golem_wd(), open = TRUE, dir_create = TRUE)
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_)
- **pkg**: The working directory. Default is `get_golem_wd()`.
- **open**: Should the file be opened once created?
- **dir_create**: Should the folder be created if it doesn’t exist?

---

**add_js_file**

Create Files

Description

These functions create files inside the inst/app folder. These functions can be used outside of a golem project.

Usage

```r
add_js_file(name, pkg = get_golem_wd(), dir = "inst/app/www",
open = TRUE, dir_create = TRUE, with_doc_ready = TRUE)
```

```r
add_js_handler(name, pkg = get_golem_wd(), dir = "inst/app/www",
open = TRUE, dir_create = TRUE)
```

```r
add_css_file(name, pkg = get_golem_wd(), dir = "inst/app/www",
open = TRUE, dir_create = TRUE)
```

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

Arguments

- **name**: The name of the module
- **pkg**: 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 file be opened?
- **dir_create**: Creates the directory if it doesn’t exist, default is TRUE.
- **with_doc_ready**: Should the default file include $( document ).ready()?
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, pkg = get_golem_wd(), open = TRUE,
            dir_create = TRUE, fct = NULL, utils = NULL, export = FALSE,
            ph_ui = " ", ph_server = " ")

Arguments
- **name** The name of the module
- **pkg** Path to the root of the package. Default is get_golem_wd().
- **open** Should the file be opened?
- **dir_create** Creates the directory if it doesn’t exist, default is TRUE.
- **fct** The name of the fct file.
- **utils** The name of the utils file.
- **export** Logical. Should the module be exported? Default is FALSE.
- **ph_ui, ph_server** Texts to insert inside the modules UI and server. For advanced use.

Note
This function will prefix the name argument with mod_.

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. By default FALSE, if TRUE display message if directory is empty.

---

**add_rstudioconnect_file**

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

Description

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

Usage

```r
add_rstudioconnect_file(pkg = get_golem_wd(), open = TRUE)
add_shinyappsio_file(pkg = get_golem_wd(), open = TRUE)
add_shinyserver_file(pkg = get_golem_wd(), open = TRUE)
```

Arguments

**pkg**  Where to put the app.R. Default is `get_golem_wd()`.

**open**  Open the file

Note

In previous versions, this function was called `add_rconnect_file`.

Examples

```r
# Add a file for Connect
if (interactive()){  
  add_rstudioconnect_file()
}

# Add a file for Shiny Server
if (interactive()){  
  add_shinyserver_file()
}

# Add a file for Shinyapps.io
if (interactive()){  
  add_shinyappsio_file()
}
```
**Amend golem config file**

**Description**
Amend golem config file

**Usage**
```r
amend_golem_config(key, value, config = "default",
pkg = get_golem_wd(), talkative = TRUE)
```

**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_NAME environment variable ("default" if the variable does not exist).
- `pkg`: Path to the root of the package. Default is `get_golem_wd()`.
- `talkative`: Should the messages be printed to the console?

---

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

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

**Usage**
```r
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/blog/a-little-trick-for-debugging-shiny/](https://rtask.thinkr.fr/blog/a-little-trick-for-debugging-shiny/) for more context.

**Usage**

```r
browser_button()
```

**Value**

Prints the code to the console.

---

**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**

```r
bundle_resources(path, app_title, name = "golem_resources", version = "0.0.1", meta = NULL, head = NULL, attachment = NULL, package = NULL, all_files = 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.
**cat_dev**

Functions already made dev dependent

**Description**

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

**Usage**

```r
cat_dev(...)  
print_dev(...)  
message_dev(...)  
warning_dev(...)  
browser_dev(...)```

**Arguments**

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

---

**create_golem**

Create a package for Shiny App using golem

**Description**

Create a package for Shiny App using golem

**Usage**

```r
create_golem(path, check_name = TRUE, open = TRUE,  
             package_name = basename(path), without_comments = FALSE, ...)```

**Arguments**

`path`  
Name of the folder to create the package in. This will also be used as the package name.

`check_name`  
When using this function in the console, you can prevent the package name from being checked.

`open`  
Boolean open the created project

`package_name`  
Package name to use. By default it’s `basename(path)` but if `path == '.'` and `package_name` not explicitly given, then `basename(getwd())` will be used.
without_comments
  Boolean start project without golem comments
  ...
  not used

fields
  Fields per level
  ...
  not used

detach_all_attached   Detach all attached package

Description
  Detach all attached package

Usage
  detach_all_attached()

document_and_reload   Document and reload your package

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

Usage
  document_and_reload(pkg = get_golem_wd())

Arguments
  pkg
    Path to the root of the package. Default is get_golem_wd().

expect_shinytag

Description
  Test helpers

Usage
  expect_shinytag(object)
  expect_shinytaglist(object)
  expect_html_equal(ui, html)
Arguments

object the object to test
ui output of an UI function
html html file to compare to ui

Value

A testthat result

Examples

expect_shinytag(shiny::tags$span("1"))
expect_shinytaglist(shiny::tagList(1))

fill_desc pkg_name, pkg_title, pkg_description, author_first_name,
author_last_name, author_email, repo_url = NULL,
pkg = get_golem_wd()

Description

Fill your description

Usage

fill_desc(pkg_name, pkg_title, pkg_description, author_first_name,
author_last_name, author_email, repo_url = NULL,
pkg = get_golem_wd())

Arguments

pkg_name The name of the package
pkg_title The title of the package
pkg_description Description of the package
author_first_name First Name of the author
author_last_name Last Name of the author
author_email Email of the author
repo_url URL (if needed)
pkg Path to look for the DESCRIPTION. Default is get_golem_wd().
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

get_sysreqs  get system requirements

Description

get system requirements

Usage

get_sysreqs(packages, quiet = TRUE, batch_n = 30)

Arguments

packages  character vector of packages names
quiet  boolean if TRUE the function is quiet
batch_n  number of simultaneous packages to ask
**golem**

A package for building Shiny App

---

**Description**


**Author(s)**

**Maintainer:** Vincent Guyader <vincent@thinkr.fr> (0000-0003-0671-9270)

Authors:

- Colin Fay <contact@colinfay.me> (0000-0001-7343-1846)
- Sébastien Rochette <sebastien@thinkr.fr> (0000-0002-1565-9313)
- Cervan Girard <cervan@thinkr.fr> (0000-0002-4816-4624)

Other contributors:

- Novica Nakov <nnovica@gmail.com> [contributor]
- ThinkR [copyright holder]

**See Also**

Useful links:

- [https://github.com/ThinkR-open/golem](https://github.com/ThinkR-open/golem)
- Report bugs at [https://github.com/ThinkR-open/golem/issues](https://github.com/ThinkR-open/golem/issues)

---

**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
**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**

```r
set_golem_options(golem_name = pkgload::pkg_name(),
                 golem_version = pkgload::pkg_version(),
                 golem_wd = pkgload::pkg_path(), app_prod = FALSE, talkative = TRUE)

set_golem_wd(path = pkgload::pkg_path(), talkative = TRUE)

set_golem_name(name = pkgload::pkg_name(), path = pkgload::pkg_path(),
               talkative = TRUE)

set_golem_version(version = pkgload::pkg_version(),
                   path = pkgload::pkg_path(), talkative = TRUE)

get_golem_wd(use_parent = TRUE, path = pkgload::pkg_path())

get_golem_name(config = Sys.getenv("R_CONFIG_ACTIVE", "default"),
               use_parent = TRUE, path = pkgload::pkg_path())

get_golem_version(config = Sys.getenv("R_CONFIG_ACTIVE", "default"),
                   use_parent = TRUE, path = pkgload::pkg_path())
```

**Arguments**

- `golem_name` Name of the current golem.
- `golem_version` Version of the current golem.
- `golem_wd` Working directory of the current golem package.
- `app_prod` Is the `{golem}` in prod mode?
- `talkative` Should the messages be printed to the console?
- `path` The path to set the golem working directory. Note that it will be passed to `normalizePath`.
- `name` The name of the app
- `version` The version of the app
- `use_parent` TRUE to scan parent directories for configuration files if the specified config file isn’t found.
- `config` Name of configuration to read from. Defaults to the value of the `R_CONFIG_NAME` environment variable ("default" if the variable does not exist).
Set Functions

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

Get Functions

Reads the information from `golem-config.yml`

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

---

**use_external_js_file**  Use Files

**Description**

These functions download files from external sources and install them inside the appropriate directory.

**Usage**

```r
use_external_js_file(url, name, pkg = get_golem_wd(),
  dir = "inst/app/www", open = TRUE, dir_create = TRUE)
```

```r
use_external_css_file(url, name, pkg = get_golem_wd(),
  dir = "inst/app/www", open = TRUE, dir_create = TRUE)
```

**Arguments**

- `url`  String representation of URL for the file to be downloaded
- `name`  The name of the module
- `pkg`  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 file be opened?
- `dir_create`  Creates the directory if it doesn’t exist, default is `TRUE`.  
use_favicon  

Add a favicon to your shinyapp

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

Usage

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

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)
- **pkg**: 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
- **ico**: path to favicon file
- **rel**: rel
- **resources_path**: prefix of the resource path of the app
- **ext**: the extension of the favicon

Examples

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

use_recommended_deps  Add recommended elements

Description

**use_recommended_deps**  Adds shiny, DT, attempt, glue, golem, htmltools to dependencies

**use_recommended_tests**  Adds a test folder and copy the golem tests
use_utils_ui

Usage

use_recommended_deps(pkg = get_golem_wd(), recommended = c("shiny", "DT", "attempt", "glue", "htmltools", "golem"))

use_recommended_tests(pkg = get_golem_wd())

Arguments

pkg Path to the root of the package. Default is get_golem_wd().

recommended A vector of recommended packages.

Description

use_utils_ui Use the utils files

use_utils_server Copies the golem_utils_server.R to the R folder.

Usage

use_utils_ui(pkg = get_golem_wd())

use_utils_server(pkg = get_golem_wd())

Arguments

pkg Path to the root of the package. Default is get_golem_wd().

golem_opts A list of Options to be added to the app

Add Golem options to a Shiny App

with_golem_options

Description

Add Golem options to a Shiny App

Usage

with_golem_options(app, golem_opts)

Arguments

app the app object.

golem_opts A list of Options to be added to the app
Value

a shiny.appObj object

Note

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

activate_js, 2
add_css_file (add_js_file), 6
add_dockerfile, 4
add_dockerfile_heroku (add_dockerfile), 4
add_dockerfile_shinyproxy (add_dockerfile), 4
add_fct, 5
add_js_file, 6
add_js_handler (add_js_file), 6
add_module, 7
add_rconnect_file
   (add_rstudioconnect_file), 8
add_resource_path, 7
add_rstudioconnect_file, 8
add_shinyappsio_file
   (add_rstudioconnect_file), 8
add_shinyserver_file
   (add_rstudioconnect_file), 8
add_ui_server_files (add_js_file), 8
add_utils (add_fct), 5
addins, 3
amend_golem_config, 9
app_dev (app_prod), 9
app_prod, 9

browser_button, 10
browser_dev (cat_dev), 11
bundle_resources, 10

cat_dev, 11
create_golem, 11
detach_all_attached, 12
document_and_reload, 12

effect_html_equal (expect_shinytag), 12
effect_shinytag, 12
effect_shinytaglist (expect_shinytag), 12
favicon (use_favicon), 18
fill_desc, 13
get_golem_name (set_golem_options), 16
get_golem_options, 14
get_golem_version (set_golem_options), 16
get_golem_wd (set_golem_options), 16
get_sysreqs, 14
go_to_app_server (addins), 3
go_to_app_ui (addins), 3
go_to_deploy (addins), 3
go_to_dev (addins), 3
go_to_run_app (addins), 3
go_to_run_dev (addins), 3
go_to_start (addins), 3
golem, 15
golem-package (golem), 15
insert_ns (addins), 3
invoke_js (activate_js), 2
make_dev, 15
message_dev (cat_dev), 11
print_dev (cat_dev), 11
remove_favicon (use_favicon), 18
resolveDependencies, 10
set_golem_name (set_golem_options), 16
set_golem_options, 16
set_golem_version (set_golem_options), 16
set_golem_wd (set_golem_options), 16
use_external_css_file
   (use_external_js_file), 17
use_external_js_file, 17
use_favicon, 18
use_recommended_deps, 18
use_recommended_tests
    (use_recommended_deps), 18
use_utils_server (use_utils_ui), 19
use_utils_ui, 19
utils::download.file, 18

warning_dev (cat_dev), 11
with_golem_options, 19