# Package ‘folders’

March 15, 2024

**Title** Standardized Folder Names

**Version** 0.1.0

**URL** `https://github.com/deohs/folders`

**BugReports** `https://github.com/deohs/folders/issues`

**Description** Supports the use of standardized folder names.

**Depends** R (>= 3.6)

**License** MIT + file LICENSE

**Encoding** UTF-8

**Imports** config (>= 0.3), here (>= 0.1), yaml (>= 2.2.1)

**RoxygenNote** 7.2.3

**Suggests** testthat (>= 2.3.2)

**NeedsCompilation** no

**Author** Brian High [aut, cre],
University of Washington [cph, fnd]

**Maintainer** Brian High `<bhigh@live.com>`

**Repository** CRAN

**Date/Publication** 2024-03-15 21:20:02 UTC

## R topics documented:

## `cleanup_folders` *Cleanup Folders*

### Description

Remove empty folders from the folders list as well as the configuration file.

### Usage

```
cleanup_folders(folders, conf_file = NULL, keep_conf = TRUE, recursive = FALSE)
```

### Arguments

| | |
|---|---|
| `folders` | (list) A named list of standard folders for an R project. |
| `conf_file` | (character) Configuration file to read/write. See: config::get(). (Default: NULL) |
| `keep_conf` | (boolean) Keep the configuration file if TRUE. (Default: TRUE) |
| `recursive` | (boolean) Cleanup subfolders recursively if TRUE. (Default: FALSE) |

### Value

(integer) A vector of results: 0 for success; 1 for failure; NULL for skipped.

### Details

Each empty folder in the list of folders will be removed. If recursive is set to TRUE, then empty subfolders will be removed first. The configuration file will be removed if keep_conf is set to FALSE.

### Examples

```
# Create list of standard folder names and store in a configuration file
conf_file <- tempfile("folders.yml")     # Using tempfile() for testing only
folders <- get_folders(conf_file)

# Testing only: Append folder names to parent folder path --
#               This would NOT be needed or desired in normal usage
folders <- lapply(folders, function(x) file.path(tempdir(), x))

# Create a folder for each item in "folders" list
result <- create_folders(folders)

# Remove empty folders, leaving only those with files or subfolders in them
result <- cleanup_folders(folders)
```

## `create_folders` *Create Folders*

### Description

Create a standardized set of folders under a parent folder of an R project.

### Usage

```
create_folders(folders, showWarnings = FALSE, recursive = TRUE)
```

### Arguments

| | |
|---|---|
| `folders` | (list) A named list of standard folders for an R project. |
| `showWarnings` | (boolean) Show warnings. See: base::dir.create(). (Default: FALSE) |
| `recursive` | (boolean) Support recursive folder creation. See: base::dir.create(). (Default: TRUE) |

### Value

(vector) A named vector for the results of "dir.create" operations.

### Details

For each folder in the "folders" list, here::here() and base::dir.create() are used to create a subfolder under the parent folder. Warnings are silenced in case the folder already exists. Recursive folder creation is supported. These two features can be controlled with the "showWarnings" and "recursive" parameters. A TRUE value in the returned vector means the folder was created by dir.create(). If a folder already exists, the returned vector will have a FALSE value for that folder.

### Examples

```
# Create list of standard folder names and store in a configuration file
conf_file <- tempfile("folders.yml")     # Using tempfile() for testing only
folders <- get_folders(conf_file)

# Testing only: Append folder names to parent folder path --
#               This would NOT be needed or desired in normal usage
folders <- lapply(folders, function(x) file.path(tempdir(), x))

# Create a folder for each item in "folders" list
result <- create_folders(folders)

# Check results
file.exists(conf_file)
sapply(folders, dir.exists)

# Create a data file and confirm that it exists
```

```
df <- data.frame(x = letters[1:3], y = 1:3)
file_path <- here::here(folders$data, "data.csv")
write.csv(df, file_path, row.names = FALSE)
file.exists(file_path)
```

---

## get_folders *Get Folders*

---

### Description

Return a named list of standard folder names. Save a config file if missing.

### Usage

```
get_folders(conf_file = NULL, conf_name = Sys.getenv("R_CONFIG_NAME"))
```

### Arguments

| | |
|---|---|
| `conf_file` | (character) Configuration file to read/write. See: config::get(). (Default: NULL) |
| `conf_name` | (character) Name of configuration to read. See: config::get(). (Default: Sys.getenv("R_CONFIG_NAME' |

### Value

(list) The named folders for a standard file structure, will be returned as a list.

### Details

The list of folders can be used to create any which are missing or to refer to a folder path by name to avoid hardcoding paths in scripts. You can refer to folders in this way to avoid the use of setwd() in scripts.

Ideally the folder paths will be subfolders relative to the parent folder and will be standard names used in several projects for consistency.

If there is a configuration file, then it will be read with config::get(). Otherwise a built-in default list will be returned. If you want to use a list from a non-default section of the configuration file, set the name of the section with the "conf_name" parameter.

### Examples

```
folders <- get_folders()                # Configuration file not used
conf_file <- tempfile("folders.yml")    # Using tempfile() for testing only
folders <- get_folders(conf_file)
folders <- get_folders(conf_file, conf_name = "custom")
Sys.setenv(R_CONFIG_NAME = "custom")
folders <- get_folders(conf_file)
```

# Index