Package 'here'

December 13, 2020

Title A Simpler Way to Find Your Files

Version 1.0.1

Date 2020-12-13

Description Constructs paths to your project's files.
Declare the relative path of a file within your project with 'i_am()'.
Use the 'here()' function as a drop-in replacement for 'file.path()',
it will always locate the files relative to your project root.

License MIT + file LICENSE

URL https://here.r-lib.org/, https://github.com/r-lib/here

BugReports https://github.com/r-lib/here/issues

Imports rprojroot (>= 2.0.2)

Suggests conflicted, covr, fs, knitr, palmerpenguins, plyr, readr,
rlang, rmarkdown, testthat, uuid, withr

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1.9000

Config/testthat/edition 3

NeedsCompilation no

Author Kirill Müller [aut, cre] (<https://orcid.org/0000-0002-1416-3412>),
Jennifer Bryan [ctb] (<https://orcid.org/0000-0002-6983-2759>)

Maintainer Kirill Müller <krlmlr+r@mailbox.org>

Repository CRAN

Date/Publication 2020-12-13 07:30:02 UTC

R topics documented:

---

dr_here *Situation report*

Description

dr_here() shows a message that by default also includes the reason why here() is set to a particular directory. Use this function if here() gives unexpected results.

Usage

```
dr_here(show_reason = TRUE)
```

Arguments

show_reason   [logical(1)]
                   Include reason in output of dr_here(), defaults to TRUE.

Examples

```
dr_here()
```

---

here *Find your files*

Description

here() uses a reasonable heuristics to find your project's files, based on the current working directory at the time when the package is loaded. Use it as a drop-in replacement for [file.path()](#), it will always locate the files relative to your project root.

Usage

```
here(...)
```

Arguments

...   [character]
                   Path components below the project root, can be empty. Each argument should
                   be a string containing one or more path components separated by a forward slash
                   "/".

Details

This package is intended for interactive use only. Use [rprojroot::has_file()](#) or the other functions in the **rprojroot** package for more control, or for package development.

If here() raises an error or otherwise behaves unexpectedly, you may have attached **plyr** or another package after **here**. Correct this using the **conflicted** package, or use here::here("data","df.rda").

Project root

The project root is established with a call to here::i_am(). Although not recommended, it can be changed by calling here::i_am() again.

In the absence of such a call (e.g. for a new project), starting with the current working directory during package load time, the directory hierarchy is walked upwards until a directory with at least one of the following conditions is found:

- contains a file .here
- contains a file matching [.]Rproj$ with contents matching ^Version: in the first line
- contains a file DESCRIPTION with contents matching ^Package:
- contains a file remake.yml
- contains a file .projectile
- contains a directory .git
- contains a file .git with contents matching ^gitdir:
- contains a directory .svn

In either case, here() appends its arguments as path components to the root directory.

Examples

```
here()
## Not run:
here("some", "path", "below", "your", "project", "root.txt")
here("some/path/below/your/project/root.txt")

## End(Not run)
```

---

i_am   *Declare location of current script or report*

Description

Add a call to here::i_am("<project-relative path>.<ext>") at the top of your R script or in the first chunk of your rmarkdown document. This ensures that the project root is set up correctly: subsequent calls to here() will refer to the implied project root. If the current working directory is outside of the project where the script or report is intended to run, it will fail with a descriptive message.

Usage

```
i_am(path, ..., uuid = NULL)
```

Arguments

path	[character(1)] The path to the current script or report, relative to the project root. Passing an absolute path raises an error.
...	Must be empty, reserved for future use.
uuid	[character(1)] **Experimental**)' If not NULL, a unique string that is matched against the first 100 lines of the file. Use uuid::UUIDgenerate() to create a unique string that can be used as a uuid argument.

Details

Relying on the project root determined with a project file, the default for versions prior to 1.0.0, only weakly protects against running a script from an arbitrary directory outside the intended project. The i_am() function offers a stronger way to define the project root: it will ensure that the project root actually contains a file in that location, optionally checking for file contents that uniquely identify the file via the nonce argument.

This function will fail if the script or report is moved within the project. Update the i_am() call to reflect the new location. If you use the nonce argument for extra safety, be sure to change it when you save an existing script or report under a new name.

Value

This function is called for its side effects.

Examples

```
## Not run:
here::i_am("prepare/penguins.R")
here::i_am("analysis/report.Rmd", uuid = "f9e884084b84794d762a535f3facec85")

## End(Not run)
```

Index