Package: audit.workbench (via r-universe)

September 16, 2024

Type Package

Title RStudio/Workbench User Acceptance Tests

Version 0.6.4

Description Testing whether data scientists can do what they expect in RStudio Workbench.

License file LICENSE

Imports audit.base (>= 0.6.9), BiocManager, cli, dplyr, fs, httr, processx, purrr, quarto (>= 1.3), remotes, rlang, rmarkdown, rstudioapi, serverHeaders, stringr, tibble, utils, withr, yaml

Suggests reticulate, testthat (>= 3.0.0)

Remotes jumpingrivers/audit.base, jumpingrivers/serverheaders

Config/testthat/edition 3

Encoding UTF-8

LazyData false

RoxygenNote 7.2.3

Repository https://jumpingrivers.r-universe.dev

RemoteUrl https://github.com/jumpingrivers/audit.workbench

RemoteRef HEAD

RemoteSha b1248bfed3aabe0a6223080a8053f45747392425

Contents

`check` *Run UAT Checks*

Description

This functions runs all UAT tests. To skip tests, set check to 'no' in the config yaml file. See 'create_config()'

Usage

```
check(server, dir = ".", debug_level = 0:2)
```

Arguments

`server`	Posit Workbench server. If NULL, use the ENV variable WORKBENCH_SERVER
`dir`	directory location of the the config file
`debug_level`	Integer, 0 to 2.

Details

Debug level description * 0: clean-up all files; suppress all noise * 1: clean-up all files, but display build steps * 2: No clean-up and display build steps

---

`check_bioconductor` *R6 classes*

Description

R6 classes

R6 classes

Details

Check classes

Super classes

`audit.base::logger` -> `audit.base::base_check` -> `check_bioconductor`

Methods

Public methods:

- check_bioconductor$check()
- check_bioconductor$clone()

Method `check()`**:** Checks that bioconductor URLs are accessible

Usage:

```
check_bioconductor$check(debug_level)
```

Arguments:

`debug_level` See check() for details

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
check_bioconductor$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`check_core_r_pkgs` *R6 classes*

Description

R6 classes

R6 classes

Details

Checks that core R packages are pre-installed as described in https://docs.posit.co/ide/server-pro/reference/r_package_depend

Super classes

`audit.base::logger` -> `audit.base::base_check` -> `check_core_r_pkgs`

Methods

Public methods:

- check_core_r_pkgs$check()
- check_core_r_pkgs$clone()

Method `check()`**:** Test for core R packages

Usage:

```
check_core_r_pkgs$check(debug_level)
```

Arguments:

debug_level See check() for details

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
check_core_r_pkgs$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

`check_cran` *R6 classes*

Description

R6 classes

R6 classes

Details

Check classes

Super classes

`audit.base::logger` -> `audit.base::base_check` -> `check_cran`

Methods

Public methods:

- `check_cran$check()`
- `check_cran$clone()`

Method `check()`**:** Installs a package from CRAN

Usage:

```
check_cran$check(debug_level)
```

Arguments:

debug_level See check() for details

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
check_cran$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

check_file_permissions *R6 classes*

Description

R6 classes

R6 classes

Details

Check classes

Super classes

audit.base::logger -> audit.base::base_check -> check_file_permissions

Methods

Public methods:

- check_file_permissions$check()
- check_file_permissions$clone()

Method `check()`**:** Check

Usage:

```
check_file_permissions$check(debug_level)
```

Arguments:

`debug_level` See check() for details

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
check_file_permissions$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

check_github *R6 classes*

Description

R6 classes

R6 classes

Details

Check classes

Super classes

audit.base::logger -> audit.base::base_check -> check_github

Methods

Public methods:

- check_github$check()
- check_github$clone()

Method `check()`**:** Checks installs a package from github#

Usage:

```
check_github$check(debug_level)
```

Arguments:

`debug_level` See check() for details

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
check_github$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

check_git_cloning *R6 classes*

Description

R6 classes

R6 classes

Details

Check classes

Super classes

audit.base::logger -> audit.base::base_check -> check_git_cloning

Methods

Public methods:

- check_git_cloning$check()
- check_git_cloning$clone()

Method `check()`**:** Checking that we can access and clone from github.com

Usage:

```
check_git_cloning$check(debug_level)
```

Arguments:

`debug_level` See check() for details

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
check_git_cloning$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

check_python_pip *R6 classes*

Description

R6 classes

R6 classes

Details

Check classes

Super classes

audit.base::logger -> audit.base::base_check -> check_python

Methods

Public methods:

- check_python_pip$check()
- check_python_pip$clone()

Method `check()`**:** Checks python pip install

Usage:

```
check_python_pip$check(debug_level)
```

Arguments:

`debug_level` See check() for details

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
check_python_pip$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

check_python_reticulate *R6 classes*

Description

R6 classes

R6 classes

Details

Check classes

Super classes

audit.base::logger -> audit.base::base_check -> check_python

Methods

Public methods:

- check_python_reticulate$check()
- check_python_reticulate$clone()

Method `check()`**:** Checks python can be used via reticulate

Usage:

```
check_python_reticulate$check(debug_level)
```

Arguments:

`debug_level` See check() for details

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
check_python_reticulate$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

check_rmd *R6 Quarto checks*

Description

A short description...

Usage

```
types
```

Format

An object of class `character` of length 3.

create_config *Create test config*

Description

This function creates an example config. By default all tests are TRUE. Some RSC servers intentionally won't have some features implemented.

Usage

```
create_config(dir = ".", default = TRUE, type = c("merge", "force", "error"))
```

Arguments

`dir`	Directory of the config file
`default`	Default value for if a test should run (logical)
`type`	Merge type, see details

Details

If a test is missing from the config file, it is assume to be TRUE. Therefore, the config file can be quite short and just list exceptions. If the config file is missing, then all tests are carried out.

When merging configs, we either * merge the new with existing * force: overwrite existing file * error: if a config file exists, raise an error

Index