Package 'rAccess'

May 9, 2026

Title Access Control Module for 'shiny' Applications

Version 0.1.1.3

Description Provides a flexible framework for implementing hierarchical access control in 'shiny' applications. Features include user permission management through a two-tier system of access panels and units, pluggable 'shiny' module for administrative interfaces, and support for multiple storage backends (local, 'AWS S3', 'Posit Connect'). The system enables fine-grained control over application features, with built-in audit trails and user management capabilities. Integrates seamlessly with 'Posit Connect's authentication system.

License Apache License (>= 2)

Encoding UTF-8

RoxygenNote 7.3.2

Imports config, dplyr, DT, here, htmlwidgets, httr, jsonlite, magrittr, pins, R6, shiny, shinyjs, shinyWidgets, tibble, tidyr, yaml, shinyalert, utils

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

URL https://johnsonandjohnson.github.io/rAccess/, https://github.com/johnsonandjohnson/rAccess/

BugReports https://github.com/johnsonandjohnson/rAccess/issues/

NeedsCompilation no

Author Peyman Eshghi [aut, cre] (ORCID: <https://orcid.org/0000-0003-1613-2705>), Nandu Krishnan [aut], Nadia Abraham [aut], Harika Adapala [ctb], Johnson & Johnson Innovative Medicine [cph, fnd]

Maintainer Peyman Eshghi <peymaan.es@gmail.com>

Repository CRAN

Date/Publication 2025-11-05 23:30:02 UTC

Contents

`get_accesshistory` *Get all access list till date*

Description

Get all access list till date

Usage

```
get_accesshistory(pin_board, pin_name)
```

Arguments

`pin_board`	Pin board
`pin_name`	Pin Name

get_accesslist *Get access list at a given date or in a specific time period*

Description

Get access list at a given date or in a specific time period

Usage

```
get_accesslist(pin_board, pin_name, datemin, datemax = NA)
```

Arguments

pin_board	Pin board
pin_name	pin name
datemin	Date in "YYYY-MM-DD" format
datemax	Date, either NA or in "YYYY-MM-DD" format

Value

A list of dataframes with access details

get_admins *Gets the list of admins*

Description

Gets the list of admins

Usage

```
get_admins(pin_board, pin_name, admin_panel)
```

Arguments

pin_board	Pin board
pin_name	pin name
admin_panel	Admin panel name

get_board *Gather pin_board elements*

Description

Gather pin_board elements

Usage

```
get_board(pin_board, pin_name)
```

Arguments

pin_board	pin board
pin_name	pin name

Value

A list with access_panels, access_units, access_df, access_list

get_granted_units *Helper function to get access units/panels with access for a user from an existing pin board*

Description

Helper function to get access units/panels with access for a user from an existing pin board

Usage

```
get_granted_units(user_id, pin_board, pin_name)
```

Arguments

user_id	user id
pin_board	pin board
pin_name	pin name

Value

A vector with access units

get_user_api *Get user data using API*

Description

Get user data using API

Usage

```
get_user_api(contact_info, url, api_key = NULL)
```

Arguments

contact_info	User entered search text
url	URL
api_key	Valid api key or NULL

Value

A tibble with user id and username

module_iam_server *Server logic of module_iam*

Description

Server logic of module_iam

Usage

```
module_iam_server(id, rAccess_obj)
```

Arguments

id	Module's ID
rAccess_obj	New instance of rAccess(R6 object)

module_iam_ui *User-interface definition of module_iam*

Description

This is a Shiny module used by the main shiny web application

Usage

```
module_iam_ui(id)
```

Arguments

id	User ID

module_sub_iam_server *Server logic of module_sub_iam*

Description

Server logic of module_sub_iam

Usage

```
module_sub_iam_server(id, access_panel_id, rAccess_obj)
```

Arguments

id	Module's ID
access_panel_id	Access panel ID
rAccess_obj	New instance of rAccess(R6 object)

module_sub_iam_ui *User-interface definition of module_sub_iam*

Description

This is a Shiny module used by the main shiny web application

Usage

```
module_sub_iam_ui(id)
```

Arguments

id	Module's ID

rAccess *rAccess module functions - R6 object*

Description

The rAccess class encapsulates various methods used in the IAM (Identity and Access Management) module. It provides functionalities to check user permissions, verify admin status, retrieve user details, and interact with pin boards for access control configuration.

Fields

`user` User ID

`app_name` Application Name

`pin_name` Pin name

`pin_list` List of pins

`pin_board` Pin board object

`access_panels` Available access panels

`access_units` Available access units

`access_mode` Access mode, e.g., "default", "single unit"

`user_df` Data frame of user IDs and names

`switch_size` Size of UI switch elements (e.g., "small", "large", "default", "mini", "small", "normal", "large")

`unit_display` Display type for access units ("switch", "dropdown")

`board_type` Type of pin board ("local", "s3", "rconnect")

`local_board_path` Local path for local pin boards

`s3_bucket` S3 bucket name

`s3_access_key` S3 access key

`s3_secret_key` S3 secret key

`use_rconnect_users` Boolean, use rconnect users in conjunction with user_df

`config` Configuration file content

`data` List of data paths from config

`verbose` Boolean, whether to print logs

`panel_config` Complete panel structure

`secure_mode` Boolean, enforce access requirement

Methods

`initialize(user, ...)` Constructor to create an instance with specified parameters.

`check_access(user_id, access_panel)` Checks user access rights for a given panel.

`is_admin()` Checks if the current user is an admin.

`no_admin()` Checks if there are no admins in the admin panel.

`get_userlist_unit(access_panel, access_unit)` Gets list of users with access to the specified unit.

`rAccessThemes()` Includes custom CSS themes for the app.

`get_user_accesslist(user_id)` Lists access units available to a user, including "everyone".

`get_superAdmins()` Returns list of admin user IDs.

Public fields

`user` User ID

`app_name` App Name

`pin_name` A field that takes the argument pin_name

`pin_list` A field that takes the argument pin_list

`pin_board` A field that takes the argument pin_board

`access_panels` A field that takes the argument access_panels

`access_units` A field that takes the argument access_units

`access_mode` Enables user to select access modes.Available access modes are : default - Allows access to multiple access panels and multiple access units, single unit - Allows access to single access unit within an access panel.

`user_df` A data.frame with user id and user name

`switch_size` Takes values : 'default', 'mini', 'small', 'normal', 'large'. Determines the size of access unit switches used in the module.

`unit_display` Takes values : 'switch', 'dropdown'. Determines the type of display for access units. Defaults to 'switch'.

`board_type` Board type. Takes values "local", "s3", "rconnect"

`local_board_path` Local board path.

`s3_bucket` S3 bucket

`s3_access_key` S3 Access Key

`s3_secret_key` S3 Secret Key

`use_rconnect_users` If true then rconnect users will be combined with the given user_df

`config` rAccess configuration file

`data` Lists all datapaths in config file

`verbose` If TRUE, prints all data base updates in the log

`panel_config` A list with entire panel structure including datapaths

`secure_mode` If TRUE, then user should have access to at least one access_unit/access_panel to use the app.

Methods

Public methods:

- rAccess$new()
- rAccess$matched_users()
- rAccess$check_access()
- rAccess$is_admin()
- rAccess$no_admin()
- rAccess$get_userlist_unit()
- rAccess$rAccessThemes()
- rAccess$get_user_accesslist()
- rAccess$get_superAdmins()
- rAccess$clone()

Method `new()`**:** Constructor to initialize an rAccess object

Usage:

```
rAccess$new(
  user = NULL,
  pin_board = NULL,
  app_name = NULL,
  pin_name = NULL,
  access_panels,
  access_units = NULL,
  access_mode = "default",
  user_df = NULL,
  switch_size = NULL,
  unit_display = "switch",
  board_type = NULL,
  local_board_path = NULL,
  s3_bucket = NULL,
  s3_access_key = NULL,
  s3_secret_key = NULL,
  use_rconnect_users = TRUE,
  config = NULL,
  verbose = FALSE,
  secure_mode = FALSE
)
```

Arguments:

`user` User ID

`pin_board` Pin board

`app_name` App name

`pin_name` Pin name

`access_panels` Access panels

`access_units` Access units

`access_mode` Access mode

`user_df` Data Frame with username and userid

`switch_size` Determines size of access unit switches : `default`, `mini`, `small`, `normal`, `large`

`unit_display` Determines the type of display for access units : `switch`, `dropdown`.

`board_type` Pin board type: `local`, `s3`, `rconnect`

`local_board_path` Local path to save pin_board when board_type is `local`.

`s3_bucket` S3 bucket id

`s3_access_key` Access key to S3 bucket

`s3_secret_key` Secret Key to S3 bucket

`use_rconnect_users` If TRUE, then rconnect users will be combined with user_df when deployed.

`config` rAccess configuration file

`verbose` If TRUE, prints all data base updates in the log

`secure_mode` If TRUE, then user should have access to at least one access_unit/access_panel to use the app.

Returns: pin_name, access_panels, access_units, Pin_board ...

Method `matched_users()`**:** Find users matching search input in self$user_df

Usage:

```
rAccess$matched_users(contact_info)
```

Arguments:

`contact_info` User entered search text

Returns: A data.frame

Method `check_access()`**:** To check user access rights to an access unit within a particular access panel. Returns access details from the app's access pin board that matches given user_id and access panel

Usage:

```
rAccess$check_access(user_id = self$user, access_panel)
```

Arguments:

`user_id` User ID

`access_panel` Access Panel name

Method `is_admin()`**:** To check if the user is Admin in order to provide access to IAM module. Returns TRUE if it is an admin user.

Usage:

```
rAccess$is_admin()
```

Method `no_admin()`**:** check if there is no user in the ADMIN panel Returns FALSE if there is one or more admins.

Usage:

```
rAccess$no_admin()
```

Method `get_userlist_unit()`**:** Gets user list filtered by given access unit

Usage:

```
rAccess$get_userlist_unit(access_panel, access_unit)
```

Arguments:

`access_panel` Access panel name

`access_unit` Access unit name

Method `rAccessThemes()`**:** Function to inline js/css into the main app's HTML

Usage:

```
rAccess$rAccessThemes()
```

Method `get_user_accesslist()`**:** Function to get list of access units for a given user. Note that it will contain access units that are accessible by `everyone`.

Usage:

```
rAccess$get_user_accesslist(user_id = self$user)
```

Arguments:

`user_id` User ID

Returns: A list

Method `get_superAdmins()`**:** Function to get app admins

Usage:

```
rAccess$get_superAdmins()
```

Returns: ADMIN user ids

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

`rconnect_pin_board` *Function to create rconnect pinboard*

Description

Function to create rconnect pinboard

Usage

```
rconnect_pin_board(server, key)
```

Arguments

`server`	rconnect server
`key`	API KEY to connect with rconnect servers

Value

a pin_board

s3_config *Set environmental variables to connect to AWS S3 bucket*

Description

Set environmental variables to connect to AWS S3 bucket

Usage

```
s3_config(access_key, secret_key, region = "us-east-1")
```

Arguments

access_key	Character. AWS access key.
secret_key	Character. AWS secret key.
region	Character. AWS region. Defaults to "us-east-1".

s3_pinboard *Function to create s3 pin_board*

Description

Function to create s3 pin_board

Usage

```
s3_pinboard(
  s3_bucket,
  s3_access_key,
  s3_secret_key,
  s3_region = "us-east-1",
  s3_prefix
)
```

Arguments

s3_bucket	s3 bucket name
s3_access_key	access key to connect to s3 bucket
s3_secret_key	secret key to connect to s3 bucket
s3_region	s3 bucket region
s3_prefix	Prefix to

Value

s3 pin board

use_config *Function to add rAccess configuration file to the given directory*

Description

Copies a configuration file (.yml) from the package's config directory to a specified path.

Usage

```
use_config(file_name = "rAccess.yml", path = getwd())
```

Arguments

file_name	Config file name with .yml extension
path	Directory to which config file is to be added

Index