

Package ‘shinydbauth’

July 23, 2025

Title Simple Authentication for 'shiny' Applications

Version 1.0.0.1

Description Provides a simple authentication mechanism for single 'shiny' applications.
Authentication and password change functionality are performed calling user provided functions that typically access some database backend.
Source code of main applications is protected until authentication is successful.

License GPL-3

URL <https://github.com/diegoefe/shinydbauth>

Encoding UTF-8

RoxygenNote 7.2.3

Imports R6, shiny, htmltools, DT (>= 0.5), openssl, R.utils,
billboardr, sscript, yaml, glue

Suggests keyring, testthat (>= 2.1.0), knitr, rmarkdown

NeedsCompilation no

Author Diego Florio [aut, cre] (ORCID:
<<https://orcid.org/0009-0002-1799-0189>>)

Maintainer Diego Florio <diegoefe@gmail.com>

Repository CRAN

Date/Publication 2023-12-10 11:30:02 UTC

Contents

custom-labels	2
fab_button	2
module-authentication	3
module-chpass	5
secure-app	6
use_language	8

Index	9
--------------	----------

custom-labels	<i>Modify shinydbauth labels to use custom text</i>
---------------	---

Description

See all labels registered with `get_labels()`, then set custom text with `set_labels()`.

Usage

```
set_labels(language, ...)  
  
get_labels(language = "en")
```

Arguments

language	Language to use for labels, supported values are : "en", "es".
...	A named list with labels to replace.

Value

`get_labels()` return a named list with all labels registered.

Examples

```
# In global.R for example:  
set_labels(  
  language = "en",  
  "Please authenticate" = "You have to login",  
  "Username:" = "What's your name:",  
  "Password:" = "Enter your password:"  
)
```

fab_button	<i>Create a FAB button</i>
------------	----------------------------

Description

Create a fixed button in page corner with additional button(s) in it

Usage

```
fab_button(
    ...,
    position = c("bottom-right", "top-right", "bottom-left", "top-left", "none"),
    animation = c("slidein", "slidein-spring", "fountain", "zoomin"),
    toggle = c("hover", "click"),
    inputId = NULL,
    label = NULL
)
```

Arguments

...	actionButtons to be used as floating buttons.
position	Position for the button.
animation	Animation when displaying floating buttons.
toggle	Display floating buttons when main button is clicked or hovered.
inputId	Id for the FAB button (act like an actionButton).
label	Label for main button.

module-authentication *Authentication module*

Description

Authentication module

Usage

```
auth_ui(
  id,
  status = "primary",
  tags_top = NULL,
  tags_bottom = NULL,
  background = NULL,
  choose_language = NULL,
  lan = NULL,
  ...
)

auth_server(
  input,
  output,
  session,
  check_credentials,
  use_token = FALSE,
  lan = NULL
)
```

Arguments

<code>id</code>	Module's id.
<code>status</code>	Bootstrap status to use for the panel and the button. Valid status are: "default", "primary", "success", "warning", "danger".
<code>tags_top</code>	A tags (div, img, ...) to be displayed on top of the authentication module.
<code>tags_bottom</code>	A tags (div, img, ...) to be displayed on bottom of the authentication module.
<code>background</code>	A optionnal css for authentication background. See example.
<code>choose_language</code>	logical/character. Add language selection on top ? TRUE for all supported languages or a vector of possibilities like c("en", "es"). If enabled, <code>input\$shinydbauth_language</code> is created
<code>lan</code>	A language object. See use_language
<code>...</code>	: Used for old version compatibility.
<code>input, output, session</code>	Standard Shiny server arguments.
<code>check_credentials</code>	Function with two arguments (user, the username provided by the user and password, his/her password). Must return a list with at least 2 (or 4 in case of sqlite) slots : • result : logical, result of authentication. • user_info : list. What you want about user ! (sqlite : the line in db corresponding to the user). • expired : logical, is user has expired ? Always FALSE if db doesn't have a expire column. Optional. • authorized : logical, is user can access to his app ? Always TRUE if db doesn't have a applications column. Optional.
<code>use_token</code>	Add a token in the URL to check authentication. Should not be used directly.

Value

A reactiveValues with 3 slots :

- **result** : logical, result of authentication.
- **user** : character, name of connected user.
- **user_info** : information about the user.

module-chpass

Change password module

Description

Change password module

Usage

```
chpass_ui(id, tag_img = NULL, status = "primary", lan = NULL)
```

```
chpass_server(
  input,
  output,
  session,
  update_credentials,
  validate_pwd = NULL,
  use_token = FALSE,
  lan = NULL
)
```

Arguments

id	Module's id.
tag_img	A tags\$img to be displayed on the authentication module.
status	Bootstrap status to use for the panel and the button. Valid status are: "default", "primary", "success", "warning", "danger".
lan	An language object. Should not be used directly.
input, output, session	Standard Shiny server arguments.
update_credentials	A function to perform an action when changing password is successful. Two arguments will be passed to the function: user (username) and password (the new password). Must return a list with at least a slot result with TRUE or FALSE, according if the update has been successful.
validate_pwd	A function to validate the password enter by the user. Default is to check for the password to have at least one number, one lowercase, one uppercase and be of length 6 at least.
use_token	Add a token in the URL to check authentication. Should not be used directly.

secure-app

*Secure a Shiny application and manage authentication***Description**

Secure a Shiny application and manage authentication

Usage

```
secure_app(
  ui,
  ...,
  head_auth = NULL,
  theme = NULL,
  language = "en",
  fab_position = "bottom-right"
)

secure_server(
  check_credentials,
  timeout = 15,
  inputs_list = NULL,
  keep_token = FALSE,
  validate_pwd = NULL,
  update_credentials = NULL,
  session = shiny::getDefaultReactiveDomain()
)

create_server(check_credentials, update_credentials, server_fn)
```

Arguments

ui	UI of the application.
...	Arguments passed to auth_ui .
head_auth	Tag or list of tags to use in the <head> of the authentication page (for custom CSS for example).
theme	Alternative Bootstrap stylesheet, default is to use readable, you can use themes provided by shinythemes. It will affect the authentication panel and the admin page.
language	Language to use for labels, supported values are : "en", "es".
fab_position	Position for the FAB button, see fab_button for options.
check_credentials	Function passed to auth_server .
timeout	Timeout session (minutes) before logout if sleeping. Default to 15. 0 to disable.

<code>inputs_list</code>	<code>list</code> . If database credentials, you can configure inputs for editing users information. See Details.
<code>keep_token</code>	Logical, keep the token used to authenticate in the URL, it allow to refresh the application in the browser, but careful the token can be shared between users ! Default to FALSE.
<code>validate_pwd</code>	A function to validate the password enter by the user. Default is to check for the password to have at least one number, one lowercase, one uppercase and be of length 6 at least.
<code>update_credentials</code>	Function passed to <code>chpass_server</code> .
<code>session</code>	Shiny session.
<code>server_fn</code>	Function that returns the authenticated server.

Details

If database credentials, you can configure inputs with `inputs_list` for editing users information from the admin console. `start`, `expire`, `admin` and `password` are not configurable. The others columns are rendering by default using a `textInput`. You can modify this using `inputs_list`. `inputs_list` must be a named list. Each name must be a column name, and then we must have the function shiny to call `fun` and the arguments `args` like this : `list(group = list( fun = "selectInput", args = list( choices = c("all", "restricted"), multiple = TRUE, selected = c("all", "restricted") ) ) )`

You can specify if you want to allow downloading users file, sqlite database and logs from within the admin panel by invoking `options("shinydbauth.download")`. It defaults to `c("db", "logs", "users")`, that allows downloading all. You can specify `options("shinydbauth.download" = "db")` if you want allow admin to download only sqlite database, `options("shinydbauth.download" = "logs")` to allow logs download or `options("shinydbauth.download" = "")` to disable all.

Using `options("shinydbauth.pwd_validity")`, you can set password validity period. It defaults to `Inf`. You can specify for example `options("shinydbauth.pwd_validity" = 90)` if you want to force user changing password each 90 days.

Using `options("shinydbauth.pwd_failure_limit")`, you can set password failure limit. It defaults to `Inf`. You can specify for example `options("shinydbauth.pwd_failure_limit" = 5)` if you want to lock user account after 5 wrong password.

`create_server` calls `secure_server` and, if authentication is ok, passes `user_info` to `server_fn`

Value

A `reactiveValues` containing informations about the user connected.

Note

A special input value will be accessible server-side with `input$shinydbauth_where` to know in which step user is : authentication, application, admin or password.

use_language	<i>Use shinydbauth labels</i>
--------------	-------------------------------

Description

See all labels registered with `get_labels()`, then set custom text with `set_labels()`.

Usage

```
use_language(lan = "en")
```

Arguments

lan Language to use for labels, supported values are : "en", "es".

Value

A language object

Examples

```
use_language(lan = "en")
```

Index

`auth_server`, [6](#)
`auth_server (module-authentication)`, [3](#)
`auth_ui`, [6](#)
`auth_ui (module-authentication)`, [3](#)

`chpass_server`, [7](#)
`chpass_server (module-chpass)`, [5](#)
`chpass_ui (module-chpass)`, [5](#)
`create_server`, [7](#)
`create_server (secure-app)`, [6](#)
`custom-labels`, [2](#)

`fab_button`, [2](#), [6](#)

`get_labels (custom-labels)`, [2](#)

`module-authentication`, [3](#)
`module-chpass`, [5](#)

`secure-app`, [6](#)
`secure_app (secure-app)`, [6](#)
`secure_server`, [7](#)
`secure_server (secure-app)`, [6](#)
`set_labels (custom-labels)`, [2](#)

`use_language`, [4](#), [8](#)