Package 'polished'

July 1, 2020

Type Package

Title Authentication and Administration for 'shiny' Apps

Version 0.1.0

Maintainer Andy Merlino <andy.merlino@tychobra.com>

Description Easily add modern authentication and user administration to your 'shiny' apps. Customize user sign in and registration pages to match your brand. Control who can access one or more of your 'shiny' apps.

License MIT + file LICENSE

URL https://github.com/tychobra/polished, https://polished.tech

BugReports https://github.com/tychobra/polished/issues

Encoding UTF-8

LazyData true

Imports apexcharter, digest, dplyr, DT, htmltools, htmlwidgets, httr, jose, jsonlite, lubridate, purrr, R6, rlang, shiny, shinycssloaders, shinydashboard, shinydashboardPlus, shinyFeedback, shinyjs, shinyWidgets, stringr, tibble, tidyr, uuid, xts

RoxygenNote 7.1.0

NeedsCompilation no

Author Andy Merlino [aut, cre], Patrick Howard [aut]

Repository CRAN

Date/Publication 2020-07-01 12:30:03 UTC

R topics documented:

default_admin_ui_options 2
email_input 2
firebase_dependencies 3
firebase_init 4

global_sessions_config 4
profile_module 6
profile_module_ui 6
providers_ui 7
secure_server 7
secure_static 8
secure_ui 9
Sessions 10
set_config_env 12
sign_in_module 12
sign_in_module_ui 13
sign_in_no_invite_module 13
sign_in_no_invite_module_ui 14
sign_in_ui_default 14
sign_out_from_shiny 15

Index **16**

`default_admin_ui_options` *Default Options for the Admin UI*

Description

This function specifies the default logos that are displayed in the "Admin Panel".

Usage

```
default_admin_ui_options()
```

Value

a list of html for branding the Admin UI

`email_input` *A 'shiny' email input*

Description

This is a replica of `shiny::textInput()` with the html input "type" attribute set to "email" rather than "text".

Usage

```
email_input(inputId, label, value = "", width = NULL, placeholder = NULL)
```

Arguments

inputId	The input slot that will be used to access the value.
label	Display label for the control, or NULL for no label.
value	Initial value.
width	The width of the input, e.g. '400px'.
placeholder	A character string giving the user a hint as to what can be entered into the control. Internet Explorer 8 and 9 do not support this option.

---

firebase_dependencies *Load the Firebase JavaScript dependencies into the UI*

Description

Under the hood, 'polished' uses Firebase JavaScript dependencies to handle user user authentication. This function loads the required Firebase JavaScript dependencies in the the UI of your 'shiny' app.

Usage

```
firebase_dependencies(services = c("auth"), firebase_version = "7.15.5")
```

Arguments

services	character vector of Firebase services to load into the UI. Valid strings are "auth", "firestore", "functions", "messaging", and "storage"
firebase_version	character string of the Firebase version. Defaults to 7.15.5.

Value

the html <script> tags for the Firebase JavaScript dependencies

Examples

```
firebase_dependencies()
```

firebase_init *Initialize Firebase*

Description

Executes a couple lines of JavaScript to initialize Firebase. This function should be called in your 'shiny' UI immediately after `firebase_dependencies`.

Usage

```
firebase_init(firebase_config)
```

Arguments

`firebase_config`
: list of firebase configuration

Value

a character string of JavaScript code to initialize Firebase

Examples

```
## Not run:
my_config <- list(
  apiKey = "your Firebase API key",
  authDomain = "your Firebase auth domain",
  projectId = "your Firebase Project ID"
)

firebase_init(my_config)

## End(Not run)
```

global_sessions_config *Configuration for global sessions*

Description

This is the primary function for configuring 'polished'. It configures your app's instance of the `Sessions` class that manages your user's 'polished' sessions. Call this function in your "global.R" file. See https://github.com/Tychobra/polished/blob/master/inst/examples/polished_example_01/global.R for a complete example.

Usage

```
global_sessions_config(
  app_name,
  api_key,
  firebase_config = NULL,
  admin_mode = FALSE,
  is_invite_required = TRUE,
  api_url = "https://api.polished.tech",
  sign_in_providers = c("google", "email")
)
```

Arguments

app_name	the name of the app.
api_key	the API key. Either from polished.tech or your on premise polished API deployment.
firebase_config	a list containing your Firebase project configuration. This list should have the following named elements: • apiKey • authDomain • projectId
admin_mode	FALSE by default. Set to TRUE to enter the polished Admin Panel without needing to register and sign in. This is useful during development for inviting the first users to your app. Make sure to set ‘admin_mode‘ to FALSE before deploying your app.
is_invite_required	TRUE by default. Whether or not to require the user to have an invite before registering/signing in
api_url	the API url. Defaults to "https://api.polished.tech".
sign_in_providers	the sign in providers to enable. Valid values are "google" "email", "microsoft", and/or "facebook". Defaults to `c("google","email")`.

Examples

```
## Not run:
# global.R

global_sessions_config(
  app_name = "<your app name>",
  api_key = "<your API key>"
)


## End(Not run)
```

profile_module *Profile Module Server*

Description

The server logic to accompany the profile_module_ui.

Usage

```
profile_module(input, output, session)
```

Arguments

input	the Shiny server input
output	the Shiny server output
session	the Shiny server session

profile_module_ui *Profile Module UI*

Description

Generates the UI for a user profile dropdown button to be used with the 'shinydashboard' package.

Usage

```
profile_module_ui(id, other_lis = NULL)
```

Arguments

id	the Shiny module id.
other_lis	additional html tags to place between the email address and the Sign out button in the user profile dropdown. This is often used to add a user "My Account" page/app where the user can set their account settings.

providers_ui *UI for the Firebase authentication providers buttons*

Description

Creates the html UI of the "Sign in with *" buttons. These buttons are only necessary if you enable social sign in via the `sign_in_providers` argument passed to `global_sessions_config`.

Usage

```
providers_ui(ns, sign_in_providers = c("google", "email"))
```

Arguments

`ns`	the 'shiny' namespace function created with `shiny::NS()`.
`sign_in_providers`	the sign in providers to enable. Valid values are "google" "email", "microsoft", and/or "facebook". Defaults to `c("google","email")`.

Value

the html UI of the "Sign in with *" buttons.

secure_server *Secure your 'shiny' app's server*

Description

This function is used to secure your 'shiny' app's server function. Make sure to pass your 'shiny' app's server function as the first argument to `secure_server()` at the bottom of your 'shiny' app's "server.R" file.

Usage

```
secure_server(
  server,
  custom_sign_in_server = NULL,
  custom_admin_server = NULL,
  allow_reconnect = FALSE
)
```

Arguments

server	A Shiny server function (e.g function(input,output,session) {})
custom_sign_in_server	Either NULL, the default, or a Shiny module server containing your custom sign in server logic.
custom_admin_server	Either NULL, the default, or a Shiny server function containing your custom admin server functionality.
allow_reconnect	argument to pass to the 'shiny' session$allowReconnect() function. Defaults to FALSE. Set to TRUE to allow reconnect with shiny-server and Rstudio Connect. Set to "force" for local testing. See https://shiny.rstudio.com/articles/reconnecting.html for more information.

---

secure_static *Secure a static html page*

Description

secure_static() can be used to secure any html page using 'polished'. It is often used to add 'polished' to ".Rmd" htmloutput and flexdashboards.

Usage

```
secure_static(html_file_path, global_sessions_config_args)
```

Arguments

html_file_path	the path the to html file. See the details for more info.
global_sessions_config_args	arguments to be passed to global_sessions_config.

Details

To secure a static html page, place the html page in a folder named "www" and call secure_static() from a file named "app.R". The file structure should look like:

- app.R
- www/
  - index.html

See an example here: https://github.com/Tychobra/polished_example_apps/tree/master/05_flex_dashboard

Value

a Shiny app object

`secure_ui` *Secure your 'shiny' UI*

Description

This function is used to secure your 'shiny' app's UI. Make sure to pass your 'shiny' app's UI as the first argument to `secure_ui()` at the bottom of your 'shiny' app's "ui.R" file.

Usage

```
secure_ui(
  ui,
  sign_in_page_ui = NULL,
  custom_admin_ui = NULL,
  custom_admin_button_ui = admin_button_ui("polished"),
  admin_ui_options = default_admin_ui_options()
)
```

Arguments

`ui`	UI of the application.
`sign_in_page_ui`	Either `NULL`, the default, or the HTML, CSS, and JavaScript to use for the UI of the Sign In page.
`custom_admin_ui`	Either `NULL`, the default, or a list of 2 elements containing custom UI to add additional 'shinydashboard' tabs to the Polished admin panel.
`custom_admin_button_ui`	Either `admin_button_ui("polished")`, the default, or your custom UI to take admins from the custom Shiny app to the Admin panel.
`admin_ui_options`	list of html elements to customize branding of the "Admin Panel". Valid list element names are "title", "sidebar_branding", and "browser_tab_icon". See `default_admin_ui_options` for an example.

Value

Secured Shiny app UI

Sessions *R6 class to track polished sessions*

Description

An instance of this class handles the 'polished' user sessions for each 'shiny' app using 'polished'. The 'shiny' developer should not need to interact with this class directly.

Methods

Public methods:

- `Sessions$config()`
- `Sessions$sign_in()`
- `Sessions$get_invite_by_email()`
- `Sessions$find()`
- `Sessions$refresh_email_verification()`
- `Sessions$set_signed_in_as()`
- `Sessions$get_signed_in_as_user()`
- `Sessions$set_inactive()`
- `Sessions$sign_out()`
- `Sessions$get_admin_mode()`
- `Sessions$clone()`

Method `config()`**:** polished Sessions configuration function

Usage:

```
Sessions$config(
  app_name,
  api_key,
  firebase_config = NULL,
  admin_mode = FALSE,
  is_invite_required = TRUE,
  api_url = "https://api.polished.tech",
  sign_in_providers = c("google", "email")
)
```

Details: This function is called via `global_sessions_config()` in global.R of all Shiny apps using polished.

Method `sign_in()`**:** verify the users Firebase JWT and store the session

Usage:

```
Sessions$sign_in(firebase_token, hashed_cookie)
```

Arguments:

`firebase_token` the Firebase JWT. This JWT is created client side (in JavaScript) via `firebase.auth()`.

`hashed_cookie` the hashed polished cookie. Used for tracking the user session. This cookie is inserted into the "polished.sessions" table if the JWT is valid.

Returns: NULL if sign in fails. If sign in is successful, a list containing the following:

- email
- email_verified
- is_admin
- user_uid
- hashed_cookie
- session_uid

Method `get_invite_by_email()`**:**

Usage:

```
Sessions$get_invite_by_email(email)
```

Method `find()`**:**

Usage:

```
Sessions$find(hashed_cookie, page)
```

Method `refresh_email_verification()`**:**

Usage:

```
Sessions$refresh_email_verification(session_uid, firebase_token)
```

Method `set_signed_in_as()`**:**

Usage:

```
Sessions$set_signed_in_as(session_uid, signed_in_as, user_uid = NULL)
```

Method `get_signed_in_as_user()`**:**

Usage:

```
Sessions$get_signed_in_as_user(user_uid)
```

Method `set_inactive()`**:**

Usage:

```
Sessions$set_inactive(session_uid, user_uid)
```

Method `sign_out()`**:**

Usage:

```
Sessions$sign_out(hashed_cookie, session_uid)
```

Method `get_admin_mode()`**:**

Usage:

```
Sessions$get_admin_mode()
```

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

set_config_env *Automatically set the config environment*

Description

Determines if the app is deployed to a server or running locally, and adjusts the config environment to "production" or "default" respectively. This function is almost always called in the "global.R" file of a shiny app immediately before the configuration in the "config.yml" is read in.

Usage

```
set_config_env(override = NULL)
```

Arguments

override	Set the environment to "default" or "production" manually. **CAUTION:** Be sure you know the difference between "default" & "production" configuration environments. Using the "production" environment will affect the database of the deployed application.

sign_in_module *Server logic for the sign in and register pages*

Description

This server logic accompanies the sign_in_module_ui.

Usage

```
sign_in_module(input, output, session)
```

Arguments

input	the Shiny input
output	the Shiny output
session	the Shiny session

sign_in_module_ui *UI for the sign in and register pages*

Description

UI for the sign in and register pages when a user invite is required to register and sign in. See sign_in_no_invite_module if you do not require your users to sign in and register to access your 'shiny' app.

Usage

```
sign_in_module_ui(id, register_link = "First time user? Register here!")
```

Arguments

id	the Shiny module id
register_link	The text that will be displayed in the link to go to the user registration page. The default is "First time user? Register here!". Set to NULL if you don't want to use the registration page.

sign_in_no_invite_module *sign_in_no_invite_module*

Description

sign_in_no_invite_module

Usage

```
sign_in_no_invite_module(input, output, session)
```

Arguments

input	the Shiny input
output	the Shiny output
session	the Shiny session

sign_in_no_invite_module_ui *Generates sign in page that does not require user invites*

Description

UI for the sign in and registration pages. `sign_in_no_invite_module` does not require your users to be invited to your 'shiny' app before registering and signing in. i.e. if you use `sign_in_no_invite_module`, anyone will be able to register and sign in to access your 'shiny' app. See `sign_in_module` if you wish to limit access to only invited users.

Usage

```
sign_in_no_invite_module_ui(id)
```

Arguments

`id`	the Shiny module id

sign_in_ui_default *Default UI styles for the sign-in pages*

Description

Default styling for the sign in and registration pages. Update the `sign_in_ui_default()` arguments with your brand and colors to quickly style the sign in and registration pages to match your brand.

Usage

```
sign_in_ui_default(
  sign_in_module = sign_in_module_ui("sign_in"),
  color = "#5ec7dd",
  company_name = "Your Brand Here",
 logo_top = tags$div(style = "width: 300px; max-width: 100%; color: #FFF;", class =
  "text-center", h1("Your", style = "margin-bottom: 0; margin-top: 30px;"), h1("Brand",
    style = "margin-bottom: 0; margin-top: 10px;"), h1("Here", style =
    "margin-bottom: 15px; margin-top: 10px;")),
 logo_bottom = tags$img(src = "polish/images/placeholder_company_logo.jpg", alt =
  "Placeholder Logo", style = "width: 200px; margin-bottom: 15px; padding-top: 15px;"),
 icon_href = "polish/images/polished_icon.png"
)
```

Arguments

sign_in_module	UI module for the sign in and registration pages.
color	hex color for the background and button.
company_name	your company name.
logo_top	html for logo to go above the sign in panel.
logo_bottom	html for the logo below the sign in panel.
icon_href	the url/path to the browser tab icon.

Value

the UI for the sign in page

---

sign_out_from_shiny *Sign our from your 'shiny' app*

Description

Call this function to sign a user out of your 'shiny' app. This function should be called inside the server function of your 'shiny' app. See https://github.com/Tychobra/polished/blob/master/inst/examples/polished_example_01/server.R For an example of this function being called after the user clicks a "Sign Out" button.

Usage

```
sign_out_from_shiny(session = shiny::getDefaultReactiveDomain())
```

Arguments

session	the Shiny session

Index

default_admin_ui_options, 2, *9*

email_input, 2

firebase_dependencies, 3, *4*
firebase_init, 4

global_sessions_config, 4, *7, 8*

profile_module, 6
profile_module_ui, *6*, 6
providers_ui, 7

secure_server, 7
secure_static, 8
secure_ui, 9
Sessions, 10
set_config_env, 12
sign_in_module, 12, *14*
sign_in_module_ui, *12*, 13
sign_in_no_invite_module, *13*, 13
sign_in_no_invite_module_ui, 14
sign_in_ui_default, 14
sign_out_from_shiny, 15