Package 'shinyNotes'

September 28, 2025

Title Shiny Module for Taking Free-Form Notes

Version 0.0.3

Description An enterprise-
targeted scalable and customizable 'shiny' module providing an easy way to incorporate free-form note taking or discussion boards into applications.
The package includes a 'shiny' module that can be included in any 'shiny' application to create a panel containing searchable, editable text broken down by section headers.
Can be used with a local 'SQLite' database, or a compatible remote database of choice.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports shinyjs, shiny, shinyWidgets, dplyr, DBI, dbplyr, RSQLite,
magrittr, stringr, markdown, rlang, utils

RoxygenNote 7.3.3

URL https://github.com/danielkovtun/shinyNotes

BugReports https://github.com/danielkovtun/shinyNotes/issues

Suggests testthat (>= 2.1.0), knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Daniel Kovtun [cre, aut]

Maintainer Daniel Kovtun <quantumfusetrader@gmail.com>

Repository CRAN

Date/Publication 2025-09-28 21:50:02 UTC

Contents

`connect_sqlite` *Connect to an SQLite database*

Description

Wrapper function to return a `SQLiteConnection` object for local development.

Usage

```
connect_sqlite(auto_disconnect = TRUE)
```

Arguments

`auto_disconnect`
: Should the connection be automatically closed when the `src` is deleted? Set to `TRUE` if you initialize the connection the call to `src_dbi()`. Pass `NA` to auto-disconnect but print a message when this happens.

Value

Returns an S4 object that inherits from `DBIConnection`. This object is used to communicate with the database engine. Under the hood, `dbConnect()` returns an object of class `SQLiteConnection`. See `dbConnect()` for more details.

Examples

```
connect_sqlite()
```

create_schema *Add schema to a SQLite database*

Description

Wrapper function to create a new schema in a SQLite database for local development.

Usage

```
create_schema(schema, con)
```

Arguments

schema	Schema name
con	A SQLiteConnection-class object, produced by dbConnect() or shinyNotes::connect_sqlite()

Value

None. Executes SQL query and returns silently.

Examples

```
con <- connect_sqlite()
create_schema(con, schema = "demo")
```

db.read_table *Read remote database tables into data frames with additional validation*

Description

Wrapper function to read table from default or custom schema, and return NA by default if an error is encountered.

Usage

```
db.read_table(con, table, schema = NA, collect = TRUE, error_value = NA)
```

Arguments

con	An object that inherits from DBIConnection-class, typically generated by dbConnect()
table	A character string specifying the DBMS table name.
schema	A character string specifying the schema in which the table is nested.
collect	A logical specifying whether the query results should be collected into memory or left as a lazy query.
error_value	Error value to return if dbReadTable() fails. Default is NA.

Value

If the SQL query executes successfully, the return value will be an object of class `tibble`. If an error is encountered, the return value will be inherited from the `error_value` argument provided (default is `NA`).

Examples

```
con <- connect_sqlite(auto_disconnect = FALSE)
dplyr::copy_to(con, iris, "df", temporary = FALSE)
db.read_table(con = con, table = 'df')
```

---

db.write_table *Write data frames to remote database tables with additional validation*

Description

Wrapper function to write data to table in default or custom schema. Returns `TRUE` if successful, `FALSE` otherwise.

Usage

```
db.write_table(
  con,
  data,
  table,
  schema = NA,
  append_only = FALSE,
  drop_overwrite = NA
)
```

Arguments

`con`	An object that inherits from `DBIConnection-class`, typically generated by `dbConnect()`
`data`	A `data.frame`, `tbl`, or other valid SQL data type containing the data to write to the database.
`table`	A character string specifying the DBMS table name.
`schema`	A character string specifying the schema in which the table is nested.
`append_only`	A logical specifying whether the operation is `INSERT` or `UPDATE`. Default of `append_only = FALSE` means execute `DELETE` on table, and update with new data.
`drop_overwrite`	A logical specifying whether the operation is `DROP` and `INSERT`. This will overwrite any existing field types.

Value

Returns TRUE if the SQL query executes successfully, FALSE otherwise.

Examples

```
connection <- connect_sqlite(auto_disconnect = FALSE)

db.write_table(con = connection, table = 'iris', data = iris)
```

---

demo_notes *Demo notes for testing* shinynote *module.*

Description

A dataset containing package functions and their titles for the shiny, shinyWidgets and dplyr packages. Formatted in a structure compatible with the shinynotes module.

Usage

```
demo_notes
```

Format

A tibble with 274 rows and 3 variables:

package package title, character class

category function name, character class

update function title, character class ...

Source

shiny help pages

shinyWidgets help pages

dplyr help pages

emojis *Demo notes for testing* shinynote *module.*

Description

A dataset containing package functions and their titles for the shiny, shinyWidgets and dplyr packages. Formatted in a structure compatible with the shinynotes module.

Usage

```
emojis
```

Format

A named list of length 2 with elements of length 1510:

name emoji name, character class

url emoji image url, character class ...

Source

GitHub emojis API

markdown_notes *Demo notes formatted with markdown for testing* shinynote *module.*

Description

A dataset containing examples of markdown syntax for including emojis, headers, and code blocks. Formatted in a structure compatible with the shinynotes module.

Usage

```
markdown_notes
```

Format

A tibble with 3 rows and 3 variables:

formatting text format type, character class

category type of markdown formatter, character class

update text with markdown syntax, character class ...

runExample *Run shinyNotes examples*

Description

Launch a example Shiny app that shows how to easily use `shinyNotes` in a Shiny app.

Run without any arguments to see a list of available example apps.

Usage

```
runExample(example)
```

Arguments

`example`	The app to launch

Value

None. Runs a demo Shiny application. This function normally does not return; interrupt R to stop the application.

Examples

```
## Only run this example in interactive R sessions
if (interactive()) {
  # List all available example apps
  runExample()

  runExample("demo")
}
```

shinynotes *Shiny notes module - server function*

Description

Server function for the `shinynotes` module.

Usage

```
shinynotes(
  input,
  output,
  session,
  group_column,
  selected_group,
  group_options,
  table_id,
  db_conn,
  category_options = NA,
  style_options = default_styles()
)
```

Arguments

input	Standard shiny input
output	Standard shiny output
session	Standard shiny session
group_column	Column in table to group and filter notes by.
selected_group	Currently selected group column value.
group_options	Group column row value options.
table_id	Named list with member 'table' and 'schema' referring to a database table containing notes.
db_conn	An object that inherits from DBIConnection-class, typically generated by dbConnect()
category_options	Category column row value options. Useful if table is empty. Default is NA (retrieved from data)
style_options	Optional named list of CSS styles to apply to note panel elements.

Details

The style_options argument contains the following default values:

- type = "paragraph"
- header
  - color = "#4b2c71"
  - style = "font-weight: bold; text-decoration: underline;"
- panel
  - status = "default"
  - background = "#fdfeff"
  - scrollY = "scroll"
  - max_height = "600px"

    - height = "100
    - padding = "4px"
    - width = "100
    - border_width = "2px"
    - border_radius = "4px"
    - border_style = "solid"
    - border_color = "#f5f5f5"
    - style = "text-align:left; margin-right:1px;"
  - paragraph_style = "margin: 0px 0px 1px;white-space: pre-wrap;"
  - bullet_style = "white-space: pre-wrap;"
  - hr_style = "margin-top:10px; margin-bottom:10px;"
  - ignoreCase = TRUE

Value

Module server component. Reactive expression containing the currently selected note data and database connection.

Examples

```
if(interactive()){
  shiny::callModule(
    module = shinynotes,
    id = "paragraph",
    style_options = shiny::reactive({
    list(
      "type" = "bullets",
      "header" = list("color" = "#ccc"),
      "panel" = list("scrollY" = TRUE)
      )
    }),
    group_column = "package",
    selected_group = shiny::reactive("shiny"),
    group_options = c("shiny", "shinyWidgets", "dplyr"),
    table_id = list(table = "scroll_demo", schema = "notes"),
    db_conn = connect_sqlite(auto_disconnect = FALSE)
  )
}
```

---

shinynotesUI *Shiny notes module - UI function*

Description

UI function for the shinynotes module.

Usage

```
shinynotesUI(id)
```

Arguments

id	An ID string that will be used to assign the module's namespace.

Value

Note module UI, containing note panel and control buttons. An HTML tag object that can be rendered as HTML using as.character().

Examples

```
if(interactive()){
 shinynotesUI(id = 'paragraph')
}
```

Index