

# Package ‘shinyChatR’

May 9, 2026

**Type** Package

**Title** R Shiny Chat Module

**Version** 1.2.0

**Description** Provides an easy-to-use module for adding a chat to a Shiny app. Allows users to send messages and view messages from other users. Messages can be stored in a database or a .rds file.

**License** GPL (>= 3)

**Encoding** UTF-8

**Imports** data.table, DBI, purrr, R6, shiny

**RoxygenNote** 7.3.1

**VignetteBuilder** knitr

**Suggests** covr, knitr, rmarkdown, RSQLite, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**URL** <https://github.com/julianschmocker/shinyChatR>,  
<https://julianschmocker.github.io/shinyChatR/>

**NeedsCompilation** no

**Author** Julian Schmocker [aut, cre, cph],  
Ivo Kwee [aut]

**Maintainer** Julian Schmocker <julian.schmocker@gmail.com>

**Repository** CRAN

**Date/Publication** 2024-05-25 10:30:02 UTC

## Contents

chat_server . . . . .	2
chat_ui . . . . .	3
CSVConnection . . . . .	3
DBConnection . . . . .	4
RDSConnection . . . . .	6
render_msg_divs . . . . .	7
render_msg_divs2 . . . . .	7
updateChatTextInput . . . . .	8

**Index****9**


---

chat_server	<i>A chat module for Shiny apps - server</i>
-------------	--

---

**Description**

Creates the server logic for the chat module, which handles adding new messages to the database or RDS file, and retrieving messages to display

**Usage**

```
chat_server(
  id,
  chat_user,
  db_connection = NULL,
  db_table_name = "chat_data",
  rds_path = NULL,
  csv_path = NULL,
  invalidateDSMillis = 1000,
  pretty = TRUE,
  nlast = 100
)
```

**Arguments**

id	The id of the module.
chat_user	The user name that should be displayed next to the message.
db_connection	A database connection object, created using the DBI package. If provided, the chat messages will be stored in a database table.
db_table_name	he name of the database table to use for storing the chat messages. If db_connection is provided, this parameter is required.
rds_path	The path to an RDS file to use for storing the chat messages. If provided, the chat messages will be stored in an RDS file.
csv_path	The path to an csv file to use for storing the chat messages. If provided, the chat messages will be stored in an csv file.
invalidateDSMillis	The milliseconds to wait before the data source is read again. The default is 1 second.
pretty	Logical that determines if the date should be displayed in a pretty format
nlast	The number of last messages to be read in and displayed

**Value**

the reactive values chat\_rv with all the chat information

---

chat_ui	<i>A chat module for Shiny apps - UI</i>
---------	--

---

### Description

Creates the user interface for the chat module, which includes a chat message display area, a text input field for entering new messages, and a send button.

### Usage

```
chat_ui(id, ui_title = "", height = "300px", width = "100%")
```

### Arguments

id	The id of the module
ui_title	The title of the chat area.
height	The height of the chat display area. Default is 300px.
width	The width of the chat display area.

---

CSVConnection	<i>CSVConnection R6 Class</i>
---------------	-------------------------------

---

### Description

CSVConnection R6 Class  
 CSVConnection R6 Class

### Details

An R6 class representing a connection to a CSV file for the chat module.

### Value

The full dataset  
 Save a message to data source

### Public fields

csv\_path The path to the CSV file.  
 nlast The number of messages to be read in and displayed.  
 Initialize the R6 Object

**Methods****Public methods:**

- [CSVConnection\\$new\(\)](#)
- [CSVConnection\\$get\\_data\(\)](#)
- [CSVConnection\\$insert\\_message\(\)](#)
- [CSVConnection\\$clone\(\)](#)

**Method new():***Usage:*

```
CSVConnection$new(csv_path, nlast = NULL)
```

*Arguments:*

csv\_path The path to the csv file.

nlast The number of messages to be read-in.

**Method get\_data():** Reads the full dataset*Usage:*

```
CSVConnection$get_data()
```

**Method insert\_message():***Usage:*

```
CSVConnection$insert_message(message, user, time)
```

*Arguments:*

message The message to be stores

user The user who entered the message

time The time when message was submitted

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

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

*Arguments:*

deep Whether to make a deep clone.

---

 DBConnection

*DBConnection R6 Class*


---

**Description**

DBConnection R6 Class

DBConnection R6 Class

**Details**

An R6 class representing a connection to a database for the chat module.

**Value**

The full dataset

Save a message to data source

**Public fields**

connection A database connection object, created using a package such as RSQLite.

table The table that contains the chat information.

Initialize the R6 Object

**Methods****Public methods:**

- [DBConnection\\$new\(\)](#)
- [DBConnection\\$get\\_data\(\)](#)
- [DBConnection\\$insert\\_message\(\)](#)
- [DBConnection\\$clone\(\)](#)

**Method new():**

*Usage:*

```
DBConnection$new(connection, table = "chat_data")
```

*Arguments:*

connection DB connection

table Table name

**Method get\_data():** Reads the full dataset

*Usage:*

```
DBConnection$get_data()
```

**Method insert\_message():**

*Usage:*

```
DBConnection$insert_message(message, user, time)
```

*Arguments:*

message The message to be stores

user The user who entered the message

time The time when message was submitted

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

---

RDSCConnection

*RDSCConnection R6 Class*

---

### Description

RDSCConnection R6 Class

RDSCConnection R6 Class

### Details

An R6 class representing a connection to a rds file for the chat module.

### Value

The full dataset

Save a message to data source

### Public fields

rds\_path The path to the rds file.

Initialize the R6 Object

### Methods

#### Public methods:

- [RDSCConnection\\$new\(\)](#)
- [RDSCConnection\\$get\\_data\(\)](#)
- [RDSCConnection\\$insert\\_message\(\)](#)
- [RDSCConnection\\$clone\(\)](#)

#### Method new():

*Usage:*

RDSCConnection\$new(rds\_path)

*Arguments:*

rds\_path The path to the rds file.

#### Method get\_data(): Reads the full dataset

*Usage:*

RDSCConnection\$get\_data()

#### Method insert\_message():

*Usage:*

RDSCConnection\$insert\_message(message, user, time)

*Arguments:*

message The message to be stores  
 user The user who entered the message  
 time The time when message was submitted

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

*Usage:*

RDSConnection\$clone(deep = FALSE)

*Arguments:*

deep Whether to make a deep clone.

---

render\_msg\_divs      *Render the messages for the chat*

---

### Description

Render the messages for the chat

### Usage

```
render_msg_divs(texts, users, act_user)
```

### Arguments

texts            a character vector with the texts  
 users            a character vector with the users  
 act\_user        a character with the current user (that is using the app)

### Value

The HTML code containing the chat messages

---

render\_msg\_divs2      *Render the messages for the chat*

---

### Description

Render the messages for the chat

### Usage

```
render_msg_divs2(texts, users, act_user, time, pretty = TRUE)
```

**Arguments**

texts	a character vector with the texts
users	a character vector with the users
act_user	a character with the current user (that is using the app)
time	a datetime object
pretty	a logical that indicates if it should simplify the date

**Value**

The HTML code containing the chat messages

---

updateChatTextInput    *A function to update the chat textInput*

---

**Description**

Updates the value of the chat textInput

**Usage**

```
updateChatTextInput(session = getDefaultReactiveDomain(), id, value)
```

**Arguments**

session	The shiny session.
id	The id of the module.
value	The new value that should be shown in the chat textInput.

# Index

chat\_server, 2

chat\_ui, 3

CSVConnection, 3

DBConnection, 4

RDSConnection, 6

render\_msg\_divs, 7

render\_msg\_divs2, 7

updateChatTextInput, 8