Package: rui (via r-universe)

August 25, 2024

Title A simple set of UI functions

Version 0.2.0

Description This package provides a wrapper around different cli and usethis functions, aiming at providing a small but consistent set of verbs to construct a simple R package UI.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports cli, rlang, usethis

Suggests knitr, pak, raylibr (>= 4.0.0.9001), reticulate, rmarkdown

RoxygenNote 7.3.1

Roxygen list(markdown = TRUE)

URL `https://rogiersbart.github.io/rui`

BugReports `https://github.com/rogiersbart/rui/issues`

VignetteBuilder knitr

Remotes jeroenjanssens/raylibr

Repository https://rogiersbart.r-universe.dev

RemoteUrl https://github.com/rogiersbart/rui

RemoteRef HEAD

RemoteSha f2b1270bcd939d4bd2f05ae110d5da0df76f5392

Contents

`conditions` *Conditions*

Description

These functions provide a way for signalling conditions. `rui::warn()` and `rui::error()` are drop-in replacements for `base::warning()` and `base::stop()`, which support **glue** strings and **cli** inline styles. ANSI colours are lost however. For retaining colours, `rui::alert()` can be used. We recommend doing so before issuing a warning or error.

Usage

```
alert(..., warn = FALSE, error = FALSE)

warn(..., .demo = FALSE)

error(..., .demo = FALSE)

stop(...)
```

Arguments

`...`	Character vectors supporting **glue** strings and **cli** inline styles.
`warn`	Logical. Should a warning be issued with the same message.
`error`	Logical. Should an error be issued with the same message.

See Also

`glue::glue()`, `cli::inline-markup`

`console` *The console API*

Description

This function provides an alternative to the verb-based API, by mapping the set of prefixes to the corresponding verbs.

Usage

```
console(..., object = NULL, levels = 1, .envir = parent.frame())
```

Arguments

`...`	Character vectors supporting **glue** strings and **cli** inline styles. This should start with a supported prefix, to map the action to one of the rui verbs. Otherwise, `rui::tell()` is used. Supported prefixes are(entitle), i (inform), v (approve), x (disapprove), ~ (begin/proceed), = (give), * (suggest), ? (ask), ! (alert), w (warn), e (stop), . (display), and $ (expose). Status bars can be resolved by providing single characters "c", "v", or "x" for clear, succeed and fail respectively.
`object`	Object to print the rui way through `rui::inspect()`.

`help_formatter` *Title*

Description

Title

Usage

```
help_formatter(object)
```

Arguments

`object`

`multi-line-feedback` *Provide multi-line feedback*

Description

These functions provide a way to provide multi-line feedback. This is typically useful for any kind of small tasks you want to provide feedback about, or longer tasks for which you need the feedback to persist in the console (*e.g.* running external code that does provide essential information (in such a case, these are useful in combination with `processx::run()` and its `stdout_line_callback` argument)) Use

- `rui::entitle()` for naming sections,
- `rui::inform()` for providing information,
- `rui::approve()` for succesful completion of a task or a positive test, and
- `rui::disapprove()` for unsuccesful completion or a negative test.

Usage

```
entitle(...)

inform(...)

approve(...)

disapprove(...)

title(...)
```

Arguments

`...`	Character vectors supporting **glue** strings and **cli** inline styles.

See Also

`glue::glue()`, `cli::inline-markup`

`object-inspection` *Object inspection*

Description

These functions provide a means to inspect the internal structure of objects, or design a UI to do so, which is consistent with the rest of the rui functionality. `rui::expose()` can be used to print information on different parts of an object (where the `.` in the prefix should be familiar to users of the magrittr pipe, and the `$` is one of the subsetting operators), whereas `rui::display()` would be more appropriate for the atomic types. `rui::inspect()` attempts to use both these functions together with `rui::entitle()` to process the output of a `utils::str()` call, and print the structure consistent with the rest of the rui functions, borrowing some style elements from `tibble:::print.tbl()` and `tibble::glimpse()`.

Usage

```
expose(..., level = 1)

display(...)

inspect(object, levels = 1)

extract(...)

show(...)
```

Arguments

`...`	Character vectors supporting **glue** strings and **cli** inline styles.
`level`	Level of indentation, typically useful for (nested) lists. Defaults to 1.
`object`	Object to print the structure of.
`levels`	Number of levels to include. Defaults to 1.

See Also

`glue::glue()`, `cli::inline-markup`

`python` *Make rui functions available in python through reticulate*

Description

Make rui functions available in python through reticulate

Usage

```
python()
```

`single-line-feedback` *Provide single-line feedback*

Description

These functions provide a way to provide single-line feedback. This is typically useful for longer tasks with different subtasks, for which there is no important information that should persist in the console for the user to refer back to (*e.g.* downloads, optimization, running an external code that doesn't output important information). Use

- `rui::begin()` to begin a task,
- `rui::proceed()` to proceed with another task,
- for ending the single-line feedback, any of
  - `rui::succeed()` for succesful completion,
  - `rui::fail()` for unsuccesful completion, and
  - `rui::clear()` to remove the feedback line.

Usage

```
begin(..., .envir = parent.frame())

proceed(..., .envir = parent.frame())

clear(.envir = parent.frame())

succeed(.envir = parent.frame())

fail(.envir = parent.frame())

update(...)

end(...)
```

Arguments

`...`	Character vectors supporting **glue** strings and **cli** inline styles.

See Also

`glue::glue()`, `cli::inline-markup`

`speaker` *Play a sound*

Description

Play a sound

Usage

```
speaker(path)
```

Arguments

`path`	Audio file name

`standard-text` *Output standard text messages*

Description

Output standard text messages

Usage

```
tell(..., .envir = parent.frame(), capture = FALSE)
```

Arguments

`...`	Character vectors supporting **glue** strings and **cli** inline styles.

See Also

`glue::glue()`, `cli::inline-markup`

`user-interaction` *User interaction*

Description

These functions provide a way to interact with the user. Use

- `rui::give()` to give the user a piece of code to be inserted elsewhere,
- `rui::suggest()` to suggest the user a thing to do,
- `rui::ask()` to ask the user a yes/no question. Note `rui::give()` does not support **cli** styles.

Usage

```
give(...)

suggest(...)

ask(..., .demo = FALSE)

copy(...)

do(...)
```

Arguments

`...`	Character vectors supporting **glue** strings and **cli** inline styles.

See Also

`glue::glue()`, `cli::inline-markup`

Index