Package 'LLMR.shiny'

July 21, 2026

Title Shared 'Shiny' Substrate for 'LLMR' Family GUIs

Version 0.1.1

Description Provides the shared shell that 'LLMR'-family graphical interfaces are built on. It includes provider and model selection, environment-variable-based API-key handling, demo and live runners, session cost accounting, error banners, comma-separated-value upload and column mapping, and display helpers for the shared diagnostics() and report() generics. Graphical interfaces for the method packages import these helpers rather than each reimplementing them, so a fix here is available to every interface that depends on it.

License MIT + file LICENSE

URL https://github.com/asanaei/LLMR.shiny

BugReports https://github.com/asanaei/LLMR.shiny/issues

Encoding UTF-8

RoxygenNote 7.3.3

Imports shiny, bslib, stats, utils

Suggests LLMR (>= 0.8.8), DT, testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Ali Sanaei [aut, cre]

Maintainer Ali Sanaei <sanaei@uchicago.edu>

Repository CRAN

Date/Publication 2026-07-21 06:30:02 UTC

Contents

`annotate_demo_result` *Mark a result as a demo result*

Description

Mark a result as a demo result

Usage

```
annotate_demo_result(x)
```

Arguments

x	A result object.

Value

x with a llmrshiny_demo attribute.

---

as_display_table *Coerce an arbitrary result to a display table*

Description

Data frames and matrices pass through (head-limited); a list yields its first data frame; anything else becomes a one-column capture of its structure.

Usage

```
as_display_table(x, max_rows = 500L)
```

Arguments

x	Any object.
max_rows	Row cap for the display.

Value

A data frame.

---

build_llm_config *Build an LLM config, falling back to a plain list without LLMR*

Description

Build an LLM config, falling back to a plain list without LLMR

Usage

```
build_llm_config(provider, model, ...)
```

Arguments

`provider, model`	Provider and model ids.
`...`	Passed to `LLMR::llm_config()` (e.g. `temperature`).

Value

An `llm_config` when LLMR is available, else a tagged list.

`build_runner` *Build a runner for a mode*

Description

Build a runner for a mode

Usage

```
build_runner(mode, responder = NULL)
```

Arguments

`mode`	`"demo"` or `"live"`.
`responder`	Optional demo responder (see `demo_runner()`).

Value

A demo runner for `"demo"`, or `NULL` for `"live"` (use the package default downstream).

`column_names_for_mapping` *Column names of a data frame, for a mapping select input*

Description

Column names of a data frame, for a mapping select input

Usage

```
column_names_for_mapping(data)
```

Arguments

`data`	A data frame.

Value

A character vector of column names.

condition_category *Error category of a caught condition*

Description

LLMR's classed errors carry their category in the condition class (`llmr_api_auth_error`, `llmr_api_rate_limit_error`, and so on); the class is read first. A plain `category` field or attribute is honored as a fallback for conditions built by other tools.

Usage

```
condition_category(e)
```

Arguments

`e`	A condition.

Value

A length-1 character category (e.g. `"auth"`, `"rate_limit"`, `"param"`, `"server"`, `"unknown"`), or `NA`.

cost_add_usage *Add realized usage to a cost record*

Description

Add realized usage to a cost record

Usage

```
cost_add_usage(state, tokens)
```

Arguments

`state`	A cost record.
`tokens`	A token-count list (see `extract_token_counts()`).

Value

The updated cost record.

cost_empty *An empty cost-accounting record*

Description

An empty cost-accounting record

Usage

```
cost_empty()
```

Value

A list with zeroed counters.

cost_set_plan *Record a planned run on a cost record*

Description

Record a planned run on a cost record

Usage

```
cost_set_plan(state, calls, label = "Next run")
```

Arguments

state	A cost record.
calls	Number of calls the next run will make.
label	A short label for the planned run.

Value

The updated cost record.

cost_tile *A value box reporting session usage*

Description

A value box reporting session usage

Usage

```
cost_tile(state)
```

Arguments

state	A cost record.

Value

A `bslib::value_box`.

demo_banner_ui *A banner announcing a demo result*

Description

A banner announcing a demo result

Usage

```
demo_banner_ui()
```

Value

A warning card.

demo_notice *Demo-result notice string*

Description

Demo-result notice string

Usage

```
demo_notice()
```

Value

The marker text stamped on every demo result.

demo_runner *A deterministic offline demo runner*

Description

Returns a function with the `.runner` contract `(experiments, ...)` that adds LLMR-shaped response columns. The per-row response text is decided by `responder`, a function `(text) -> character`. Results are marked as demo.

Usage

```
demo_runner(
  responder = NULL,
  text_cols = c("text", "unit", "document", "prompt", "input")
)
```

Arguments

`responder`	A function mapping a single input text to a response string. Defaults to echoing a short stub.
`text_cols`	Candidate column names to read the input text from.

Value

A runner function of class `llmrshiny_demo_runner`.

diagnostics_table *Render an object's* diagnostics() *as a display table*

Description

Render an object's diagnostics() as a display table

Usage

```
diagnostics_table(x, ...)
```

Arguments

x	A result object with an LLMR::diagnostics() method.
...	Passed to diagnostics().

Value

A data frame, or NULL when no method applies.

extract_token_counts *Extract call/token counts from a result frame*

Description

Extract call/token counts from a result frame

Usage

```
extract_token_counts(x, fallback_calls = 0L)
```

Arguments

x	A result data frame (or list containing one).
fallback_calls	Call count to assume when none can be read.

Value

A list list(calls, sent, received, total).

github_remote_for *GitHub remote for an LLMR-family package*

Description

GitHub remote for an LLMR-family package

Usage

```
github_remote_for(package)
```

Arguments

package	Package name.

Value

A "owner/repo" string for `remotes::install_github()`.

install_guidance_ui *Install-guidance card for a missing package*

Description

Install-guidance card for a missing package

Usage

```
install_guidance_ui(package, title = package)
```

Arguments

package	Package name.
title	Card title (defaults to the package name).

Value

A `bslib::card` with an `install_github()` snippet.

is_auth_error *Is a caught condition an auth error?*

Description

Is a caught condition an auth error?

Usage

```
is_auth_error(e)
```

Arguments

e	A condition.

Value

`TRUE` when the condition is an LLMR `llmr_api_auth_error` or its category otherwise resolves to `"auth"`.

is_demo_result *Is a result a demo result?*

Description

Is a result a demo result?

Usage

```
is_demo_result(x)
```

Arguments

x	A result object.

Value

`TRUE` when marked by `annotate_demo_result()`.

key_state *Key state for a provider, read from the environment only*

Description

Never returns the key value; only whether one was found and in which variable.

Usage

```
key_state(provider)
```

Arguments

provider	Provider id.

Value

A list: `provider`, `display`, `env_vars`, `found`, `env_var`.

key_state_tile *A tile reporting key state (never the key value)*

Description

A tile reporting key state (never the key value)

Usage

```
key_state_tile(state)
```

Arguments

state	A `key_state()` list.

Value

A `bslib` value box or warning card.

live_run_blocker_ui *A banner shown when a live run is blocked for want of a key*

Description

A banner shown when a live run is blocked for want of a key

Usage

```
live_run_blocker_ui(state)
```

Arguments

state	A key_state() list.

Value

A warning card.

llmr_error_banner *Map a caught LLM error to a banner card*

Description

An auth failure becomes a key-state banner naming the environment variables to set; any other error shows its message verbatim. Neither crashes the app.

Usage

```
llmr_error_banner(e, provider = NULL)
```

Arguments

e	A caught condition.
provider	Optional provider id, for the key banner.

Value

A bslib::card.

map_columns *Map user columns to* text *(and optionally* labels*)*

Description

Map user columns to `text` (and optionally `labels`)

Usage

```
map_columns(data, text_col, label_col = NULL, keep_original = TRUE)
```

Arguments

`data`	A data frame.
`text_col`	Name of the column to become `text`.
`label_col`	Optional name of the column to become `labels`.
`keep_original`	Keep the original columns alongside the mapped ones. A pre-existing `text` (or `labels`) column that is not itself the mapped source is preserved under a `.original` suffix rather than overwritten.

Value

A data frame with a `text` column (and `labels` when requested), always with one row per input row.

null-coalesce *Null-coalescing operator*

Description

Returns `x` unless it is `NULL` or empty, in which case `y`.

Usage

```
x %||% y
```

Arguments

`x, y`	Values; `x` is preferred when non-empty.

Value

`x` if non-`NULL` and length > 0, otherwise `y`.

persona_selector_server *Server for the persona selector module*

Description

Renders a compact overview of `data` as a multi-select table and returns the selected row indices (relative to `data`) as a reactive. When `data` carries the persona contract (see `LLMR::llm_persona_overview()`), the overview columns are chosen automatically; otherwise the first few columns are shown.

Usage

```
persona_selector_server(
  id,
  data,
  overview = NULL,
  page_length = 8L,
  height = "260px"
)
```

Arguments

`id`	Module id.
`data`	A persona data frame (or a `reactive` returning one).
`overview`	Optional overview data frame, or a function `function(df)` building one. Defaults to `LLMR::llm_persona_overview()` when LLMR is available, else the first columns of `data`.
`page_length`	Rows per page in the table. Default `8`.
`height`	CSS height for the scrollable table body. Default `"260px"`.

Value

A `reactive` returning an integer vector of selected row indices into `data` (`integer(0)` when nothing is selected). When `DT` is not installed the module renders nothing and the reactive is always `integer(0)`, matching the install guidance shown by `persona_selector_ui()`.

See Also

`persona_selector_ui()`.

persona_selector_ui *UI for the persona selector module*

Description

Renders the selectable persona table. Pair with `persona_selector_server()`. The table's height is set by the `height` argument of `persona_selector_server()`, which controls the scrollable body.

Usage

```
persona_selector_ui(id)
```

Arguments

`id`	Module id.

Value

A Shiny UI element (a `DT` output).

See Also

`persona_selector_server()`.

pkg_available *Is a package installed?*

Description

Is a package installed?

Usage

```
pkg_available(package)
```

Arguments

`package`	Package name.

Value

`TRUE` if the package can be loaded.

provider_choices *Provider choices for a select input*

Description

Provider choices for a select input

Usage

```
provider_choices()
```

Value

A named character vector (`display` -> `provider`).

provider_default_model *Default model for a provider*

Description

Default model for a provider

Usage

```
provider_default_model(provider)
```

Arguments

`provider`	Provider id.

Value

The default model string, or `""`.

provider_display_name *Display name for a provider*

Description

Display name for a provider

Usage

```
provider_display_name(provider)
```

Arguments

provider	Provider id.

Value

The display name, or the provider id.

provider_env_vars *Environment-variable names a provider's key may live in*

Description

Environment-variable names a provider's key may live in

Usage

```
provider_env_vars(provider)
```

Arguments

provider	Provider id.

Value

A length-2 character vector `c("<P>_API_KEY", "<P>_KEY")`.

provider_registry *Provider registry*

Description

Provider registry

Usage

```
provider_registry()
```

Value

A data frame of `provider`, `display`, and `default_model`.

read_csv_path *Read a CSV from a path*

Description

Read a CSV from a path

Usage

```
read_csv_path(path)
```

Arguments

`path`	File path.

Value

A data frame.

read_csv_upload *Read an uploaded CSV (a Shiny fileInput value)*

Description

Read an uploaded CSV (a Shiny fileInput value)

Usage

```
read_csv_upload(file)
```

Arguments

file	A fileInput value (a list with datapath).

Value

A data frame.

report_text *Render an object's* report() *prose, falling back to print output*

Description

Render an object's report() prose, falling back to print output

Usage

```
report_text(x, ...)
```

Arguments

x	A result object with an LLMR::report() method, or any object.
...	Passed to report().

Value

A character scalar of report text.

safe_llmr_call *Run an expression, capturing any error as a banner*

Description

Evaluates `expr`; on error returns a list with `ok = FALSE` and a ready-made `ui` banner (auth-aware) instead of stopping. On success returns `ok = TRUE` and the `value`.

Usage

```
safe_llmr_call(expr, provider = NULL)
```

Arguments

`expr`	An expression to evaluate.
`provider`	Optional provider id, for an auth banner.

Value

`list(ok, value, error, ui)`.

shell_context *Wire the standard sidebar and return the shared reactive context*

Description

Call once at the top of a GUI's server with the top-level `input`, `output`, `session`. It keeps the model field in sync with the provider, renders the key and cost tiles, tracks usage, and returns a list of reactives and mutators (`provider`, `model`, `mode`, `key`, `can_run`, `set_plan`, `add_usage`) for the per-package modules to consume.

Usage

```
shell_context(input, output, session)
```

Arguments

`input`, `output`, `session`	The top-level Shiny server arguments.

Value

A shared-context list.

shell_sidebar *The standard GUI sidebar: provider, model, mode, key tile, cost tile*

Description

The standard GUI sidebar: provider, model, mode, key tile, cost tile

Usage

```
shell_sidebar(id = NULL, default_provider = "groq")
```

Arguments

id	The module namespace (or NULL for top-level inputs).
default_provider	Provider selected initially.

Value

A bslib::sidebar.

validate_column_mapping *Validate a column mapping*

Description

Validate a column mapping

Usage

```
validate_column_mapping(data, text_col, label_col = NULL)
```

Arguments

data	A data frame.
text_col	Required text column name.
label_col	Optional label column name.

Value

TRUE, or an error.

Index