Package 'corteza'

May 8, 2026

Title AI Agent Runtime

Version 0.6.3

Description An agent runtime that gives Large Language Models (LLMs) from 'Anthropic' <https://www.anthropic.com/>, 'OpenAI' <https://openai.com/>, 'Moonshot' <https://www.moonshot.ai/>, and 'Ollama' <https://ollama.com/> direct access to a live R session with managed workspace state. Tools execute as R function calls with provenance tracking, and a deterministic retrieval system keeps relevant objects in context across turns. Three entry points: a shell command-line interface (CLI), a console read-eval-print-loop via chat(), and a Model Context Protocol (MCP) server via serve() for external clients.

License Apache License (>= 2)

URL https://github.com/cornball-ai/corteza

BugReports https://github.com/cornball-ai/corteza/issues

Depends R (>= 4.4.0)

Imports callr, codetools, curl, jsonlite, llm.api, printify, processx, saber

Suggests fortunes, mx.api, simplermarkdown, tinytest

VignetteBuilder simplermarkdown

SystemRequirements On Windows, Rtools45 (R 4.5.x) or Rtools44 (R 4.4.x) is recommended so the 'bash' shell tool is available; minimal installs fall back to a 'cmd' tool. 'git' is required for the git_status, git_diff, and git_log tools (install Git for Windows, or 'pacman -Sy git' from an Rtools shell).

Encoding UTF-8

NeedsCompilation no

Author Troy Hernandez [aut, cre] (ORCID: <https://orcid.org/0009-0005-4248-604X>), cornball.ai [cph]

Maintainer Troy Hernandez <troy@cornball.ai>

Repository CRAN

Date/Publication 2026-05-04 19:00:13 UTC

Contents

`add_observer` *Add a tool-call observer to a session*

Description

Observers run after every tool call (run, denied, or declined). They receive a single `event` list with fields:

- `call` — the call list passed to `policy()`.
- `decision` — the policy decision for the call.
- `outcome` — one of `"ran"`, `"deny"`, `"declined"`.
- `result` — the string returned to the LLM.

- success — logical; TRUE only for "ran" with no tool error.
- elapsed_ms — wall time including policy overhead.
- turn_number — the session's tool-call counter.

Errors raised inside an observer are swallowed.

Usage

```
add_observer(session, observer)
```

Arguments

session	A session environment from new_session.
observer	A function of one argument (the event list).

Value

The session, invisibly.

---

chat *Start Interactive Chat*

Description

Run a conversational agent inside your R session. Tools execute as direct function calls, no MCP server needed.

Usage

```
chat(provider = NULL, model = NULL, tools = NULL, session = NULL,
     max_turns = NULL)
```

Arguments

provider	LLM provider: "anthropic", "openai", "moonshot", or "ollama". Defaults to config value or "anthropic".
model	Model name. Defaults to config value or provider default.
tools	Character vector of tool names or categories to enable. Categories: file, code, r, data, web, git, chat, memory. Use "core" for file+code+git, "all" for everything (default).
session	Session resume control. NULL (default) starts fresh, TRUE resumes the latest session, or a character session key to resume a specific session.
max_turns	Integer or NULL. Maximum LLM turns per user prompt before the loop stops with [Max turns reached]. NULL (default) reads getOption("corteza.max_turns"), then falls back to the session_setup default (50).

Value

The session object (invisibly).

Examples

```
if (interactive()) {
    # Start chatting with defaults from config
    chat()

    # Use a specific provider/model
    chat(provider = "ollama", model = "llama3.2")

    # Minimal tools for focused work
    chat(tools = "core")
}
```

`default_local_model` *Detect the preferred local Ollama model*

Description

Walks `getOption("corteza.local_models")` (default `c("gpt-oss:120b", "gpt-oss:20b")`) and returns the first one that is currently installed in the local Ollama server. Returns NULL if Ollama is unreachable or none of the candidates are installed. Cached per R process.

Usage

```
default_local_model()
```

Value

Character scalar model name, or NULL.

Examples

```
# NULL when Ollama isn't running locally; a model name otherwise.
model <- default_local_model()
is.null(model) || is.character(model)
```

`install_cli` *Install corteza CLI*

Description

Install the `corteza` command-line tool to a directory in your PATH. On Unix (Linux, macOS) installs the Rscript shebang binary. On Windows installs a `.cmd` wrapper alongside the script so `corteza` works from cmd.exe / PowerShell.

Usage

```
install_cli(path = NULL, force = FALSE)
```

Arguments

`path`	Directory to install to. Default is `~/bin` on Unix, `tools::R_user_dir("corteza", "data")/bin` on Windows.
`force`	Overwrite existing installation.

Details

Requires:

- `r` (littler) for fast R script execution (Unix only — Windows uses `Rscript`).
- The `llm.api` package for LLM connectivity
- The `corteza` package itself

After installation, run `corteza` from any terminal (you may need to add the install directory to PATH; the function prints the PATH hint if it isn't already there).

Value

The installed script path, invisibly.

Examples

```
## Not run:
install_cli()
install_cli("/usr/local/bin")

## End(Not run)
```

matrix_archive_all *Flush all in-memory matrix sessions to the pensar vault*

Description

Walks the per-room session registry and archives any turns that haven't been ingested yet via the pensar archive ingest. Each session tracks an `ingested_through` watermark so repeated calls only write new turns. Silent no-op when `pensar` is not installed.

Usage

```
matrix_archive_all(sessions, mx_sess = NULL)
```

Arguments

`sessions`	A registry environment built by `matrix_run`/`matrix_poll`. Keys are room IDs, values are session lists carrying `$history`.
`mx_sess`	Optional Matrix session for room-name lookups. When NULL, the room ID is used as the source identifier.

Value

Integer count of rooms ingested, invisibly.

matrix_configure *Configure the Matrix channel for this host*

Description

Logs in to a Matrix homeserver as the bot account, joins (or records) the target room, and writes credentials to `tools::R_user_dir("corteza", "config")/matrix.json` with file mode 0600. Call once per host. Model, provider, tools_filter, and auto_approve_asks are defaults the poll loop uses unless overridden at call time. Pre-CRAN releases stored the file at `~/.corteza/matrix.json`; that path is still read for backward compatibility, but the next `matrix_configure()` call writes to the new location.

Usage

```
matrix_configure(server, user, password, room, model = NULL,
                 provider = c("anthropic", "openai", "moonshot", "ollama"),
                 tools_filter = NULL, auto_approve_asks = FALSE)
```

Arguments

server	Character. Homeserver base URL.
user	Character. Bot localpart or full Matrix ID.
password	Character. Bot password. Stored locally so the bot can re-authenticate if its access token is invalidated.
room	Character. Room ID or alias the bot should read and post to. If the bot has been invited but not joined, it will be joined.
model	Character or NULL. Default model name.
provider	Character. LLM provider: "anthropic", "openai", "moonshot", or "ollama".
tools_filter	Character vector or NULL. Passed to `get_tools()` to restrict which tools the bot can invoke. NULL allows all registered tools.
auto_approve_asks	Logical. When TRUE, tool calls that policy returns `"ask"` for are auto-approved. Suitable for a personal bot on a trusted tailnet. When FALSE (default) asks are declined until the thumbs-up reaction protocol lands.

Value

The saved configuration, invisibly.

---

matrix_poll *One iteration of sync-and-reply*

Description

Fetches new messages across all joined rooms and runs `turn` against each. Auto-joins any pending invites the bot has received. Replies are sent back to the originating room. On first run there is no saved sync token, so this call establishes a baseline and returns without processing history. Pass `sessions = NULL` (the default) for a stateless one-shot — each incoming message builds a fresh session. Pass a registry created by `matrix_new_session_registry()` so a long-running `matrix_run` keeps a separate history per room (conversations in different rooms don't cross-contaminate).

Usage

```
matrix_poll(system = NULL, model = NULL, provider = NULL, tools_filter = NULL,
            timeout = 0L, sessions = NULL)
```

Arguments

system	Character or NULL. System prompt override.
model	Character or NULL. Model override.
provider	Character or NULL. Provider override.
tools_filter	Character vector or NULL. Tool filter override.
timeout	Integer. Long-poll timeout in milliseconds. 0 returns immediately.
sessions	Environment from `matrix_new_session_registry()` keyed by room_id, or NULL to build fresh sessions each call.

Value

An integer count of messages replied to, invisibly.

`matrix_request_flush` *Ask the running matrix bot to archive sessions to pensar*

Description

Drops an `archive.signal` file in the corteza state directory. The next iteration of the long-poll loop in `matrix_run` picks it up, runs `matrix_archive_all`, and removes the file. Safe to call from any process or scheduler — systemd, Task Scheduler, launchd, cron, or a separate R session — without needing to know the bot's PID or share its memory.

Usage

```
matrix_request_flush()
```

Value

The signal file path, invisibly.

`matrix_run` *Run the Matrix adapter as a long-poll loop*

Description

Creates one session up front and reuses it across polls so conversation history accumulates within the process lifetime. Intended as the entry point for a systemd user unit.

Usage

```
matrix_run(timeout = 30000L, system = NULL, model = NULL, provider = NULL,
           tools_filter = NULL)
```

Arguments

`timeout`	Integer. Long-poll timeout in milliseconds.
`system`	Character or NULL. System prompt override.
`model`	Character or NULL. Model override.
`provider`	Character or NULL. Provider override.
`tools_filter`	Character vector or NULL. Tool filter override.

Value

Never returns under normal operation. Crashes on fatal error so systemd can restart.

matrix_send *Send a message to a Matrix room*

Description

Send a message to a Matrix room

Usage

```
matrix_send(text, room_id = NULL, msgtype = "m.text")
```

Arguments

text	Character. Plain text body.
room_id	Character. Matrix room id. Defaults to cfg$room_id from the saved Matrix config (see matrix_configure).
msgtype	Character. Matrix msgtype, default "m.text".

Value

The event ID of the sent message.

mcp_tool_executor *Build a tool executor that routes through an MCP connection*

Description

Returns a closure suitable for the tool_executor argument of turn. Each tool call is forwarded to the connected MCP server via llm.api::mcp_call.

Usage

```
mcp_tool_executor(conn)
```

Arguments

conn	An open MCP connection (from llm.api::mcp_connect).

Value

A function with signature function(name, args) that returns an MCP-format result list.

`new_session` *Create a new turn session*

Description

Returns an environment with sensible defaults. Adapters set channel- specific fields (e.g. `approval_cb`, `tools_filter`) before calling `turn`.

Usage

```
new_session(channel = c("cli", "console", "matrix"), history = NULL,
            model_map = NULL, provider = "anthropic", tools_filter = NULL,
            system = NULL, approval_cb = NULL, max_turns = 10L, verbose = FALSE)
```

Arguments

`channel`	Character, one of "cli", "console", "matrix".
`history`	List of prior messages, or NULL.
`model_map`	Named list with `cloud` and `local` model names. Defaults to configured defaults.
`provider`	LLM provider passed to `llm.api::agent`.
`tools_filter`	Character vector passed to `get_tools()`.
`system`	System prompt override (NULL for built-in default).
`approval_cb`	Function called when policy returns `"ask"`. Signature: `function(call, decision) -> TRUE|FALSE`. Default denies (safe fallback).
`max_turns`	Maximum LLM turns per call.
`verbose`	Print tool call progress.

Value

An environment holding the session state.

Examples

```
# Build a stateless session for the CLI channel without making any
# network calls. The returned environment carries history, the
# active provider/model, and the approval callback.
s <- new_session(channel = "cli", provider = "anthropic")
is.environment(s)
identical(s$provider, "anthropic")
```

observer_progress *Built-in progress observer that prints to stdout*

Description

Prints one line per tool call suitable for an interactive REPL: `" [tool] hint (N lines)\n"`. The hint is a short summary of the call (file path, code snippet, search pattern) computed by `tool_hint()`.

Usage

```
observer_progress()
```

Value

A function to pass to `add_observer`.

policy *Evaluate policy for a tool call*

Description

Returns a decision `list(model, approval, reason)`. `model` is `"cloud"` or `"local"`; `approval` is `"allow"`, `"ask"`, or `"deny"`.

Usage

```
policy(call)
```

Arguments

`call`	A list describing the tool call. See the file header in `R/policy.R` for the expected fields.

Value

A decision list with fields `model`, `approval`, `reason`.

`serve` *Start MCP Server*

Description

Start the corteza MCP server. This exposes R tools to MCP clients like Claude Desktop, VS Code, or the corteza CLI.

Usage

```
serve(port = NULL, cwd = NULL, tools = NULL)
```

Arguments

`port`	Port number for socket transport. If NULL, uses stdio transport.
`cwd`	Working directory for the server. Defaults to current directory.
`tools`	Character vector of tools or categories to enable. Categories: file, code, r, data, web, git, chat. Use "core" for file+code+git, "all" for everything (default).

Details

The server supports two transport modes:

- **stdio** (default): For Claude Desktop and other MCP clients. Communication happens via stdin/stdout.

- **socket**: For the corteza CLI and R clients. Listens on a TCP port.

Tools Provided

- ‘read_file‘, ‘write_file‘, ‘replace_in_file‘, ‘list_files‘, ‘grep_files‘ - File operations - ‘run_r‘ - Execute R code in the server session - ‘bash‘ - Run shell commands - ‘r_help‘ - Query package docs via saber (exports, function help) - ‘installed_packages‘ - List installed packages - ‘web_search‘ - Search the web via Tavily (requires TAVILY_API_KEY) - ‘fetch_url‘ - Fetch web content - ‘git_status‘, ‘git_diff‘, ‘git_log‘ - Git operations - ‘chat‘, ‘chat_models‘ - LLM chat (requires llm.api)

Value

NULL (runs until interrupted or client disconnects)

Examples

```
## Not run:
# For Claude Desktop (stdio)
serve()

# For corteza CLI (socket) with all tools
serve(port = 7850)
```

```
# Minimal tools for small context models
serve(port = 7850, tools = "core")

# Specific categories
serve(port = 7850, tools = c("file", "git"))

## End(Not run)
```

`session_setup` *Configure and construct a session for any channel*

Description

Performs pre-turn setup common to all channels:

1. Loads project + global corteza config from `cwd`.
2. Resolves provider, model, and verifies the required API environment variable is set.
3. Registers built-in skills and loads user/project skills and skill docs from `tools::R_user_dir("corteza", "data")/skills` and `<cwd>/.corteza/skills`.
4. Loads skill packages declared in the config.
5. Optionally builds the system prompt via `load_context(cwd)`.
6. Returns a `new_session()` built from the above.

Usage

```
session_setup(channel = c("cli", "console", "matrix"), cwd = getwd(),
              provider = NULL, model = NULL, tools = NULL, system = NULL,
              approval_cb = NULL, history = NULL, load_project_context = TRUE,
              validate_api_key = TRUE, verbose = FALSE, max_turns = 50L)
```

Arguments

`channel`	Character, one of `"cli"`, `"console"`, `"matrix"`.
`cwd`	Working directory. Defaults to the current directory.
`provider`	Character or NULL. LLM provider override. NULL falls back to `config$provider`, then `"anthropic"`.
`model`	Character or NULL. Model override. NULL falls back to `config$model`, then the provider default.
`tools`	Character vector, NULL, or the string `"all"`. Tool filter passed through to `get_tools()`. NULL is treated as `"all"`.
`system`	Character or NULL. System prompt. NULL auto-builds via `load_context(cwd)` when `load_project_context = TRUE`, otherwise left NULL (channel supplies its own).
`approval_cb`	Function or NULL. Approval callback for `"ask"` verdicts; see `new_session`.

history	List or NULL. Prior conversation messages to seed the session with (each entry a list with role and content).
load_project_context	Logical. When TRUE, auto-call load_context(cwd) to assemble the system prompt. Channels with their own short system prompt (like matrix) pass FALSE.
validate_api_key	Logical. When TRUE, error if the provider's API key env var is unset or empty.
verbose	Logical. Passed through to new_session.
max_turns	Integer. Passed through to new_session. Defaults to 50, a safety net for interactive channels where a multi-step request (read + edit + verify several files) can easily exceed the new_session() default of 10.

Value

A session environment from new_session, with an extra cwd field set.

---

skill_install *Install a skill from a path or URL*

Description

Install a skill from a path or URL

Usage

```
skill_install(source, target_dir = NULL, force = FALSE)
```

Arguments

source	Path to skill directory or URL
target_dir	Installation directory. Default is tools::R_user_dir("corteza", "data")/skills.
force	Overwrite if exists

Value

Installed skill name

skill_list_installed *List installed skills*

Description

List installed skills

Usage

```
skill_list_installed(skill_dir = NULL)
```

Arguments

skill_dir	Skills directory

Value

Data frame with skill info

skill_remove *Remove an installed skill*

Description

Remove an installed skill

Usage

```
skill_remove(name, skill_dir = NULL)
```

Arguments

name	Skill name
skill_dir	Skills directory

Value

Invisible TRUE on success

skill_test *Run skill tests*

Description

Executes test_*.R files in a skill directory.

Usage

```
skill_test(path, verbose = TRUE)
```

Arguments

path	Path to skill directory
verbose	Print test output

Value

List with passed, failed, errors

subagent_kill *Kill a subagent.*

Description

Kill a subagent.

Usage

```
subagent_kill(id)
```

Arguments

id	Subagent ID.

Value

Invisible TRUE if killed, FALSE if not found.

subagent_list *List active subagents.*

Description

List active subagents.

Usage

```
subagent_list()
```

Value

List of subagent info objects.

subagent_query *Query a subagent.*

Description

Sends a prompt to a running subagent. Inside the child it runs through [turn()] with the child's persistent turn session: the LLM replies, any tool calls it makes resolve against the child's in-process skill registry, and history accumulates across queries.

Usage

```
subagent_query(id, prompt, timeout = 60L)
```

Arguments

id	Subagent ID.
prompt	Prompt to send.
timeout	Timeout in seconds (currently advisory; callr-level hard timeouts are future work).

Value

Reply text (character).

subagent_spawn *Spawn a subagent.*

Description

Starts a fresh 'callr::r_session' with corteza loaded and its tool registry set up. Stores the handle in the package-level registry keyed by subagent id.

Usage

```
subagent_spawn(task, model = NULL, tools = NULL, parent_session = NULL,
               config = NULL)
```

Arguments

task	Task description (stored for bookkeeping; not yet fed into an agent loop — see TODO on subagent_query).
model	Optional model override (reserved for later use).
tools	Optional tool filter (character vector).
parent_session	Parent session object; read for nested-spawning control and session-key derivation.
config	Config list.

Value

Subagent ID (character).

turn *Run one agent turn*

Description

Sends `prompt` to the configured LLM with tool use enabled. Every tool call the LLM makes is routed through `policy` before being dispatched. Tool dispatch is pluggable via `tool_executor`. The default is an in-process dispatcher that calls the local skill registry — suitable for `chat()` and matrix adapters running in the same R process as their skills. Pass `mcp_tool_executor` (or any `function(name, args) -> MCP-format result`) to run tools in a separate process, which is how the CLI talks to `serve()`.

Usage

```
turn(prompt, session, tool_executor = NULL, tools = NULL)
```

Arguments

prompt	Character. User prompt.
session	A session environment created by new_session.
tool_executor	Function or NULL. Dispatcher with signature function(name, args) -> list. NULL uses the in-process call_skill path.
tools	List or NULL. Tool schemas to pass the LLM. NULL uses the in-process skill registry (filtered by session$tools_filter). Pass explicit schemas when running against a remote skill source.

Value

A list with reply (character) and session (the updated session environment; also mutated in place).

---

uninstall_cli *Uninstall corteza CLI*

Description

Remove the corteza command-line tool.

Usage

```
uninstall_cli(path = NULL)
```

Arguments

path	Directory where corteza is installed. Default matches install_cli(): ~/bin on Unix, tools::R_user_dir("corteza", "data")/bin on Windows.

Value

TRUE if removed, FALSE if not found, invisibly.

Examples

```
## Not run:
uninstall_cli()

## End(Not run)
```

Index