Package: prefab (via r-universe)

May 29, 2026

Title Opinionated Project Scaffolding via Composable Themes

Version 0.4.0

Description Provides a composable theme system for project scaffolding. Themes are ordered lists of steps that deploy files, inline text, or execute functions, with per-file merge strategies for handling pre-existing content. Includes pre-set themes for R analysis projects, targets workflows, and Claude Code agent configuration.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

URL https://economic.github.io/prefab/, https://github.com/Economic/prefab

BugReports https://github.com/Economic/prefab/issues

Config/testthat/edition 3

Imports cli, fs, glue, jsonlite, rlang, rprojroot, withr

Suggests knitr, rmarkdown, rstudioapi, testthat (>= 3.0.0), yaml

VignetteBuilder knitr

Config/Needs/website economic/epitemplate

Config/pak/sysreqs cmake make libuv1-dev

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

Date/Publication 2026-03-30 17:00:02 UTC

RemoteUrl https://github.com/economic/prefab

RemoteRef HEAD

RemoteSha c0ec4323b31ec7cea79fb66197c26fe9f5e5d940

Contents

`claude_r_analysis` *Claude Code configuration theme for R analysis projects*

Description

Creates a theme that deploys Claude Code agent settings and rules for an R analysis project.

Usage

```
claude_r_analysis(settings_json = TRUE)
```

Arguments

`settings_json`	Logical. If `TRUE` (default), merges the package `settings.json` into `.claude/settings.json`.

Value

A `prefab_theme` object.

Examples

```
claude_r_analysis()
```

claude_r_package *Claude Code configuration theme for R packages*

Description

Creates a theme that deploys Claude Code agent settings and rules for an R package.

Usage

```
claude_r_package()
```

Value

A `prefab_theme` object.

Examples

```
claude_r_package()
```

claude_r_targets *Claude Code configuration theme for R targets projects*

Description

Creates a theme that deploys Claude Code agent settings and rules for an R targets project.

Usage

```
claude_r_targets(settings_json = TRUE)
```

Arguments

`settings_json`	Logical. If `TRUE` (default), merges the package `settings.json` into `.claude/settings.json`.

Value

A `prefab_theme` object.

Examples

```
claude_r_targets()
```

create_project *Create a new project and apply a theme*

Description

Creates a new project directory and applies a theme to it. If RStudio is available, opens the new project in a new session.

Usage

```
create_project(path, theme)
```

Arguments

path	Path for the new project directory. Resolved to an absolute path via `fs::path_abs()`.
theme	A `prefab_theme` object created by `new_theme()` or a pre-set theme function.

Value

The normalized project path (invisibly).

Examples

```
## Not run:
create_project("~/projects/my-analysis", r_analysis())
create_project("my-targets-project", r_targets() + claude_r_targets())

## End(Not run)
```

from_dir *Create a step-builder from a local directory*

Description

Returns a step-builder function that resolves source paths relative to a local directory. The directory path is resolved to an absolute path at creation time.

Usage

```
from_dir(path)
```

Arguments

path	Path to the directory. Resolved to absolute via `fs::path_abs()` at creation time. Must exist.

Value

A function with signature `function(source, dest, strategy = "overwrite", data = NULL)` that returns a `step_file()` object.

---

from_package *Create a step-builder from an installed package*

Description

Returns a step-builder function that resolves source paths relative to a package's `inst/` directory. Works with both installed packages and during development with `devtools::load_all()`.

Usage

```
from_package(package)
```

Arguments

`package`	Package name (string).

Value

A function with signature `function(source, dest, strategy = "overwrite", data = NULL)` that returns a `step_file()` object.

---

load_themes *Load custom theme definitions*

Description

Sources an R file containing custom theme functions into the global environment, making them available for use with `use_theme()` and `create_project()`.

Usage

```
load_themes(file = NULL)
```

Arguments

`file`	Path to an R file defining theme functions. If `NULL`, falls back to the `PREFAB_THEMES` environment variable, then `~/.prefab-themes.R`.

Details

The file is resolved in order: the explicit `file` argument, then the `PREFAB_THEMES` environment variable, then `~/.prefab-themes.R`.

Value

The file path that was sourced (invisibly), or `NULL` invisibly if no file was found.

Examples

```
## Not run:
# Source from default locations
load_themes()

# Source from a specific file
load_themes("~/my-org-themes.R")

## End(Not run)
```

`new_theme` *Create a theme from steps*

Description

Constructs a theme from step objects. `NULL` arguments are silently dropped, enabling conditional steps via `if (cond) step(...)`.

Usage

```
new_theme(...)
```

Arguments

`...`	Step objects (`step_file()`, `step_text()`, `step_run()`).

Value

A list with class `"prefab_theme"` containing a `steps` element.

`r_analysis` *R analysis project theme*

Description

Creates a theme that scaffolds a simple R analysis project with `main.R`, `README.md`, and `.gitignore`.

Usage

```
r_analysis(data_dirs = TRUE)
```

Arguments

data_dirs	Logical. If TRUE (default), creates directories ./data_raw and ./data_processed.

Value

A prefab_theme object.

Examples

```
r_analysis()
```

---

r_targets *R targets project theme*

Description

Creates a theme that scaffolds an R targets project with _targets.R, packages.R, README.md, and .gitignore.

Usage

```
r_targets()
```

Value

A prefab_theme object.

Examples

```
r_targets()
```

---

step_file *Create a file deployment step*

Description

Creates a step that deploys a file or directory to the project.

Usage

```
step_file(source, dest, strategy = "overwrite", data = NULL)
```

Arguments

source	Absolute path to the source file or directory. Typically produced by a source helper (from_package(), from_dir()) rather than written by hand.
dest	Path to the destination, relative to the project root.
strategy	How to handle a pre-existing destination file. One of "overwrite", "skip", "union", "append", or "merge_json".
data	NULL (default) for static file copy, "auto" to interpolate using only auto-discovered project variables (project_dir, package_name, year, date), or a named list of variables to interpolate into the file via {{var}} syntax before deploying.

Value

A list with class "prefab_step_file".

---

step_run *Create a function execution step*

Description

Creates a step that executes an R function for its side effects.

Usage

```
step_run(fn, ..., .label = NULL)
```

Arguments

fn	A function to execute.
...	Additional arguments passed to fn at execution time.
.label	Optional label for display. When NULL (default), captured via deparse(substitute(fn)).

Value

A list with class "prefab_step_run".

step_text *Create an inline text deployment step*

Description

Creates a step that deploys inline text content to the project. Like `step_file()` but takes a character vector instead of a source file path.

Usage

```
step_text(content, dest, strategy = "overwrite")
```

Arguments

`content`	Character vector of lines to deploy (one element per line). `character(0)` is allowed except with `strategy = "merge_json"`.
`dest`	Path to the destination, relative to the project root.
`strategy`	How to handle a pre-existing destination file. One of `"overwrite"`, `"skip"`, `"union"`, `"append"`, or `"merge_json"`.

Value

A list with class `"prefab_step_text"`.

theme_code *Print R code that reproduces a theme*

Description

Prints the R code that would reproduce the given theme via `cat()`, and returns the code invisibly as a single character string.

Usage

```
theme_code(theme)
```

Arguments

`theme`	A `prefab_theme` object.

Value

The generated R code as a single character string (invisibly).

theme_from_dir *Create a theme from a directory of template files*

Description

Converts a directory tree into a theme where each file becomes a `step_file()`. An optional `_prefab.yml` sidecar file in the directory can override the strategy and template data per file.

Usage

```
theme_from_dir(path, strategy = "skip")
```

Arguments

`path`	Path to a directory of template files. Resolved to an absolute path via `fs::path_abs()`. Must exist.
`strategy`	Default merge strategy for all files. Can be overridden per file via a `_prefab.yml` sidecar. One of `"overwrite"`, `"skip"`, `"union"`, `"append"`, or `"merge_json"`.

Value

A `prefab_theme` object.

Examples

```
## Not run:
# Create a theme from a directory of config files
use_theme(theme_from_dir("~/my-template"))

# Compose with other themes
use_theme(r_analysis() + theme_from_dir("~/my-extras"))

## End(Not run)
```

use_theme *Apply a theme to the current project*

Description

Discovers the project root and executes the theme against it.

Usage

```
use_theme(theme)
```

Arguments

theme	A prefab_theme object created by new_theme() or a pre-set theme function.

Value

The project root path (invisibly).

Examples

```
## Not run:
use_theme(r_analysis())
use_theme(r_analysis() + claude_r_analysis())

## End(Not run)
```

Index