Package 'tarchives'

June 22, 2025

Title Make Your 'targets' Pipelines into a Package

Version 0.1.1

Description Runs 'targets' pipeline in '/inst/tarchives' and stores the results
in the R user directory. This means that the user does not have to run the
process repeatedly, and the developer has the flexibility to update the data
as versions are updated.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports callr, fs, rlang, targets, usethis, withr

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 4.0.0)

NeedsCompilation no

Author Mizuki Uchida [aut, cre] (ORCID:
<https://orcid.org/0009-0003-6534-6942>)

Maintainer Mizuki Uchida <uchidamizuki@vivaldi.net>

Repository CRAN

Date/Publication 2025-06-22 04:40:02 UTC

Contents

_____

tar_archive	*Function factory for archived targets*

_____

Description

Function factory for archived targets

Usage

```
tar_archive(
  f,
  package,
  pipeline,
  envir = parent.frame(),
  script = targets::tar_config_get("script"),
  store = targets::tar_config_get("store")
)
```

Arguments

f	A function of targets package.
package	A scalar character of the package name.
pipeline	A scalar character of the pipeline name.
envir	An environment, where to run the target R script (default: _targets.R) if callr_function is NULL. Ignored if callr_function is anything other than NULL. callr_function should only be NULL for debugging and testing purposes, not for serious runs of a pipeline, etc.
	The envir argument of [tar_make()](#) and related functions always overrides the current value of tar_option_get("envir") in the current R session just before running the target script file, so whenever you need to set an alternative envir, you should always set it with tar_option_set() from within the target script file. In other words, if you call tar_option_set(envir = envir1) in an interactive session and then tar_make(envir = envir2, callr_function = NULL), then envir2 will be used.
script	Character of length 1, path to the target script file. Defaults to tar_config_get("script"), which in turn defaults to _targets.R. When you set this argument, the value of tar_config_get("script") is temporarily changed for the current function call. See [tar_script()](#), [tar_config_get()](#), and [tar_config_set()](#) for details about the target script file and how to set it persistently for a project.
store	Character of length 1, path to the targets data store. Defaults to tar_config_get("store"), which in turn defaults to _targets/. When you set this argument, the value of tar_config_get("store") is temporarily changed for the current function call. See [tar_config_get()](#) and [tar_config_set()](#) for details about how to set the data store path persistently for a project.

Value

A function.

---

tar_archive_script   *Path to the archived target script file*

Description

Path to the archived target script file

Usage

```
tar_archive_script(
  package,
  pipeline,
  envir = parent.frame(),
  script = targets::tar_config_get("script")
)
```

Arguments

package	A scalar character of the package name.
pipeline	A scalar character of the pipeline name.
envir	An environment, where to run the target R script (default: _targets.R) if callr_function is NULL. Ignored if callr_function is anything other than NULL. callr_function should only be NULL for debugging and testing purposes, not for serious runs of a pipeline, etc.
	The envir argument of [tar_make()](#) and related functions always overrides the current value of tar_option_get("envir") in the current R session just before running the target script file, so whenever you need to set an alternative envir, you should always set it with tar_option_set() from within the target script file. In other words, if you call tar_option_set(envir = envir1) in an interactive session and then tar_make(envir = envir2, callr_function = NULL), then envir2 will be used.
script	Character of length 1, path to the target script file. Defaults to tar_config_get("script"), which in turn defaults to _targets.R. When you set this argument, the value of tar_config_get("script") is temporarily changed for the current function call. See [tar_script()](#), [tar_config_get()](#), and [tar_config_set()](#) for details about the target script file and how to set it persistently for a project.

Value

A scalar character of the path to the archived target script file.

---

tar_archive_store   *Path to the archived target store directory*

Description

Path to the archived target store directory

Usage

```
tar_archive_store(package, pipeline, store = targets::tar_config_get("store"))
```

Arguments

package	A scalar character of the package name.
pipeline	A scalar character of the pipeline name.
store	Character of length 1, path to the targets data store. Defaults to tar_config_get("store"), which in turn defaults to _targets/. When you set this argument, the value of tar_config_get("store") is temporarily changed for the current function call. See [tar_config_get()](#) and [tar_config_set()](#) for details about how to set the data store path persistently for a project.

Value

A scalar character of the path to the archived target store directory.

---

tar_make_archive   *Run an archived pipeline of targets.*

Description

Run an archived pipeline of targets.

Usage

```
tar_make_archive(
  package,
  pipeline,
  names = NULL,
  shortcut = targets::tar_config_get("shortcut"),
  reporter = "silent",
  seconds_meta_append = targets::tar_config_get("seconds_meta_append"),
  seconds_meta_upload = targets::tar_config_get("seconds_meta_upload"),
  seconds_reporter = targets::tar_config_get("seconds_reporter"),
  seconds_interval = targets::tar_config_get("seconds_interval"),
  callr_function = callr::r,
```

```
    callr_arguments = targets::tar_callr_args_default(callr_function, reporter),
    envir = parent.frame(),
    script = targets::tar_config_get("script"),
    store = targets::tar_config_get("store"),
    garbage_collection = NULL,
    use_crew = targets::tar_config_get("use_crew"),
    terminate_controller = TRUE,
    as_job = targets::tar_config_get("as_job")
)
```

Arguments

package	A scalar character of the package name.
pipeline	A scalar character of the pipeline name.
names	Names of the targets to run or check. Set to NULL to check/run all the targets (default). The object supplied to names should be a tidyselect expression like any_of() or starts_with() from tidyselect itself, or tar_described_as() to select target names based on their descriptions.
shortcut	Logical of length 1, how to interpret the names argument. If shortcut is FALSE (default) then the function checks all targets upstream of names as far back as the dependency graph goes. shortcut = TRUE increases speed if there are a lot of up-to-date targets, but it assumes all the dependencies are up to date, so please use with caution. It relies on stored metadata for information about upstream dependencies. shortcut = TRUE only works if you set names.
reporter	A scalar character of the reporter type. By default, "silent". See targets::tar_make() for more options.

seconds_meta_append

Positive numeric of length 1 with the minimum number of seconds between saves to the local metadata and progress files in the data store. his is an aggressive optimization setting not recommended for most users: higher values generally make the pipeline run faster, but unsaved work (in the event of a crash) is not up to date. When the pipeline ends, all the metadata and progress data is saved immediately, regardless of seconds_meta_append.

When the pipeline is just skipping targets, the actual interval between saves is max(1, seconds_meta_append) to reduce overhead.

seconds_meta_upload

Positive numeric of length 1 with the minimum number of seconds between uploads of the metadata and progress data to the cloud (see https://books.ropensci.org/targets/cloud-storage.html). Higher values generally make the pipeline run faster, but unsaved work (in the event of a crash) may not be backed up to the cloud. When the pipeline ends, all the metadata and progress data is uploaded immediately, regardless of seconds_meta_upload.

seconds_reporter

Deprecated on 2025-03-31 (targets version 1.10.1.9010).

seconds_interval

Deprecated on 2023-08-24 (targets version 1.2.2.9001). Use seconds_meta_append and seconds_meta_upload instead.

callr_function   A function from callr to start a fresh clean R process to do the work. Set to
                   NULL to run in the current session instead of an external process (but restart
                   your R session just before you do in order to clear debris out of the global
                   environment). callr_function needs to be NULL for interactive debugging,
                   e.g. tar_option_set(debug = "your_target"). However, callr_function
                   should not be NULL for serious reproducible work.

callr_arguments

                   A list of arguments to callr_function.

envir   An environment, where to run the target R script (default: _targets.R) if
                   callr_function is NULL. Ignored if callr_function is anything other than
                   NULL. callr_function should only be NULL for debugging and testing pur-
                   poses, not for serious runs of a pipeline, etc.

                   The envir argument of [tar_make()](#) and related functions always overrides the
                   current value of tar_option_get("envir") in the current R session just before
                   running the target script file, so whenever you need to set an alternative envir,
                   you should always set it with tar_option_set() from within the target script
                   file. In other words, if you call tar_option_set(envir = envir1) in an inter-
                   active session and then tar_make(envir = envir2, callr_function = NULL),
                   then envir2 will be used.

script   Character of length 1, path to the target script file. Defaults to tar_config_get("script"),
                   which in turn defaults to _targets.R. When you set this argument, the value of
                   tar_config_get("script") is temporarily changed for the current function
                   call. See [tar_script()](#), [tar_config_get()](#), and [tar_config_set()](#) for de-
                   tails about the target script file and how to set it persistently for a project.

store   Character of length 1, path to the targets data store. Defaults to tar_config_get("store"),
                   which in turn defaults to _targets/. When you set this argument, the value
                   of tar_config_get("store") is temporarily changed for the current function
                   call. See [tar_config_get()](#) and [tar_config_set()](#) for details about how to
                   set the data store path persistently for a project.

garbage_collection

                   Deprecated. Use the garbage_collection argument of [tar_option_set()](#)
                   instead to run garbage collection at regular intervals in a pipeline, or use the
                   argument of the same name in [tar_target()](#) to activate garbage collection for
                   a specific target.

use_crew   Logical of length 1, whether to use crew if the controller option is set in
                   tar_option_set() in the target script (_targets.R). See https://books.ropensci.org/targets/crew.html for details.

terminate_controller

                   Logical of length 1. For a crew-integrated pipeline, whether to terminate the
                   controller after stopping or finishing the pipeline. This should almost always be
                   set to TRUE, but FALSE combined with callr_function = NULL will allow you to
                   get the running controller using tar_option_get("controller") for debug-
                   ging purposes. For example, tar_option_get("controller")$summary()
                   produces a worker-by-worker summary of the work assigned and completed,
                   tar_option_get("controller")$queue is the list of unresolved tasks, and
                   tar_option_get("controller")$results is the list of tasks that completed
                   but were not collected with pop(). You can manually terminate the controller

with `tar_option_get("controller")$summary()` to close down the dispatcher and worker processes.

as_job   TRUE to run as an RStudio IDE / Posit Workbench job, if running on RStudio IDE / Posit Workbench. FALSE to run as a `callr` process in the main R session (depending on the `callr_function` argument). If `as_job` is TRUE, then the `rstudioapi` package must be installed.

Value

NULL except if `callr_function = callr::r_bg()`, in which case a handle to the `callr` background process is returned. Either way, the value is invisibly returned.

---

tar_read_archive   *Read a target's value from archive storage*

Description

Read a target's value from archive storage

Usage

```
tar_read_archive(
  name,
  package,
  pipeline,
  branches = NULL,
  meta = NULL,
  store = targets::tar_config_get("store")
)

tar_read_archive_raw(
  name,
  package,
  pipeline,
  branches = NULL,
  meta = NULL,
  store = targets::tar_config_get("store")
)
```

Arguments

name   Name of the target to read. `tar_read()` expects an unevaluated symbol for the name argument, whereas `tar_read_raw()` expects a character string.

package   A scalar character of the package name.

pipeline   A scalar character of the pipeline name.

branches   Integer of indices of the branches to load if the target is a pattern.

meta	Data frame of metadata from [tar_meta()](). tar_read() with the default arguments can be inefficient for large pipelines because all the metadata is stored in a single file. However, if you call [tar_meta()]() beforehand and supply it to the meta argument, then successive calls to tar_read() may run much faster.
store	Character of length 1, path to the targets data store. Defaults to tar_config_get("store"), which in turn defaults to _targets/. When you set this argument, the value of tar_config_get("store") is temporarily changed for the current function call. See [tar_config_get()]() and [tar_config_set()]() for details about how to set the data store path persistently for a project.

Value

The target's return value from its file in _targets/objects/, or the paths to the custom files and directories if format = "file" was set.

---

tar_source_archive   *Run archived R scripts.*

Description

Run archived R scripts.

Usage

```
tar_source_archive(
  package,
  files = "R",
  envir = targets::tar_option_get("envir"),
  change_directory = FALSE
)
```

Arguments

package	A scalar character of the package name.
files	Character vector of file and directory paths to look for R scripts to run. Paths must either be absolute paths or must be relative to the current working directory just before the function call.
envir	Environment to run the scripts. Defaults to tar_option_get("envir"), the environment of the pipeline.
change_directory	
	Logical, whether to temporarily change the working directory to the directory of each R script before running it.

Value

NULL (invisibly)

---

tar_target_archive   *Declare a target to read an archive.*

Description

Declare a target to read an archive.

Usage

```
tar_target_archive(
  name,
  package,
  pipeline,
  name_archive = NULL,
  ...,
  pattern = NULL,
  packages = targets::tar_option_get("packages"),
  library = targets::tar_option_get("library"),
  deps = NULL,
  string = NULL,
  format = targets::tar_option_get("format"),
  repository = targets::tar_option_get("repository"),
  iteration = targets::tar_option_get("iteration"),
  error = targets::tar_option_get("error"),
  memory = targets::tar_option_get("memory"),
  garbage_collection = isTRUE(targets::tar_option_get("garbage_collection")),
  deployment = targets::tar_option_get("deployment"),
  priority = targets::tar_option_get("priority"),
  resources = targets::tar_option_get("resources"),
  storage = targets::tar_option_get("storage"),
  retrieval = targets::tar_option_get("retrieval"),
  cue = targets::tar_option_get("cue"),
  description = targets::tar_option_get("description")
)

tar_target_archive_raw(
  name,
  package,
  pipeline,
  name_archive = name,
  ...,
  pattern = NULL,
  packages = targets::tar_option_get("packages"),
  library = targets::tar_option_get("library"),
  deps = NULL,
  string = NULL,
  format = targets::tar_option_get("format"),
```

```
  repository = targets::tar_option_get("repository"),
  iteration = targets::tar_option_get("iteration"),
  error = targets::tar_option_get("error"),
  memory = targets::tar_option_get("memory"),
  garbage_collection = isTRUE(targets::tar_option_get("garbage_collection")),
  deployment = targets::tar_option_get("deployment"),
  priority = targets::tar_option_get("priority"),
  resources = targets::tar_option_get("resources"),
  storage = targets::tar_option_get("storage"),
  retrieval = targets::tar_option_get("retrieval"),
  cue = targets::tar_option_get("cue"),
  description = targets::tar_option_get("description")
)
```

Arguments

name
: Symbol, name of the target. In `tar_target()`, name is an unevaluated symbol, e.g. `tar_target(name = data)`. In `tar_target_raw()`, name is a character string, e.g. `tar_target_raw(name = "data")`.

  A target name must be a valid name for a symbol in R, and it must not start with a dot. Subsequent targets can refer to this name symbolically to induce a dependency relationship: e.g. `tar_target(downstream_target, f(upstream_target))` is a target named `downstream_target` which depends on a target `upstream_target` and a function `f()`.

  In most cases, The target name is the name of its local data file in storage. Some file systems are not case sensitive, which means converting a name to a different case may overwrite a different target. Please ensure all target names have unique names when converted to lower case.

  In addition, a target's name determines its random number generator seed. In this way, each target runs with a reproducible seed so someone else running the same pipeline should get the same results, and no two targets in the same pipeline share the same seed. (Even dynamic branches have different names and thus different seeds.) You can recover the seed of a completed target with `tar_meta(your_target, seed)` and run `tar_seed_set()` on the result to locally recreate the target's initial RNG state.

package
: A scalar character of the package name.

pipeline
: A scalar character of the pipeline name.

name_archive
: A scalar character of a name of archived target. If NULL, the name of the target is used. By default, NULL.

...
: Arguments to pass to targets::tar_outdated, targets::tar_make or tar_read_archive_raw.

pattern
: Code to define a dynamic branching branching for a target. In `tar_target()`, pattern is an unevaluated expression, e.g. `tar_target(pattern = map(data))`. In `tar_target_raw()`, command is an evaluated expression, e.g. `tar_target_raw(pattern = quote(map(data)))`.

  To demonstrate dynamic branching patterns, suppose we have a pipeline with numeric vector targets x and y. Then, `tar_target(z, x + y, pattern = map(x,`

	y)) implicitly defines branches of z that each compute x[1] + y[1], x[2] + y[2], and so on. See the user manual for details.
packages	Character vector of packages to load right before the target runs or the output data is reloaded for downstream targets. Use tar_option_set() to set packages globally for all subsequent targets you define.
library	Character vector of library paths to try when loading packages.
deps	Optional character vector of the adjacent upstream dependencies of the target, including targets and global objects. If NULL, dependencies are resolved automatically as usual. The deps argument is only for developers of extension packages such as tarchetypes, not for end users, and it should almost never be used at all. In scenarios that at first appear to requires deps, there is almost always a simpler and more robust workaround that avoids setting deps.
string	Optional string representation of the command. Internally, the string gets hashed to check if the command changed since last run, which helps targets decide whether the target is up to date. External interfaces can take control of string to ignore changes in certain parts of the command. If NULL, the strings is just deparsed from command (default).
format	Optional storage format for the target's return value. With the exception of format = "file", each target gets a file in _targets/objects, and each format is a different way to save and load this file. See the "Storage formats" section for a detailed list of possible data storage formats.
repository	Character of length 1, remote repository for target storage. Choices:

- "local": file system of the local machine.
- "aws": Amazon Web Services (AWS) S3 bucket. Can be configured with a non-AWS S3 bucket using the endpoint argument of [tar_resources_aws()](), but versioning capabilities may be lost in doing so. See the cloud storage section of [https://books.ropensci.org/targets/data.html]() for details for instructions.
- "gcp": Google Cloud Platform storage bucket. See the cloud storage section of [https://books.ropensci.org/targets/data.html]() for details for instructions.
- A character string from [tar_repository_cas()]() for content-addressable storage.

Note: if repository is not "local" and format is "file" then the target should create a single output file. That output file is uploaded to the cloud and tracked for changes where it exists in the cloud. As of targets version 1.11.0 and higher, the local file is no longer deleted after the target runs.

iteration	Character of length 1, name of the iteration mode of the target. Choices:

- "vector": branching happens with vctrs::vec_slice() and aggregation happens with vctrs::vec_c().
- "list", branching happens with [[]] and aggregation happens with list().
- "group": dplyr::group_by()-like functionality to branch over subsets of a non-dynamic data frame. For iteration = "group", the target must not by dynamic (the pattern argument of [tar_target()]() must be left NULL). The target's return value must be a data frame with a special tar_group

column of consecutive integers from 1 through the number of groups. Each
integer designates a group, and a branch is created for each collection of
rows in a group. See the [`tar_group()`](#) function to see how you can create
the special tar_group column with dplyr::group_by().

error   Character of length 1, what to do if the target stops and throws an error. Options:

- "stop": the whole pipeline stops and throws an error.
- "continue": the whole pipeline keeps going.
- "null": The errored target continues and returns NULL. The data hash is
  deliberately wrong so the target is not up to date for the next run of the
  pipeline. In addition, as of targets version 1.8.0.9011, a value of NULL is
  given to upstream dependencies with error = "null" if loading fails.
- "abridge": any currently running targets keep running, but no new targets
  launch after that.
- "trim": all currently running targets stay running. A queued target is al-
  lowed to start if:
    1. It is not downstream of the error, and
    2. It is not a sibling branch from the same [`tar_target()`](#) call (if the error
       happened in a dynamic branch).

  The idea is to avoid starting any new work that the immediate error impacts.
  error = "trim" is just like error = "abridge", but it allows potentially
  healthy regions of the dependency graph to begin running. (Visit https://books.ropensci.org/targets/debugging.html to learn how to debug
  targets using saved workspaces.)

memory   Character of length 1, memory strategy. Possible values:

- "auto" (default): equivalent to memory = "transient" in almost all cases.
  But to avoid superfluous reads from disk, memory = "auto" is equivalent
  to memory = "persistent" for for non-dynamically-branched targets that
  other targets dynamically branch over. For example: if your pipeline has
  tar_target(name = y, command = x, pattern = map(x)), then tar_target(name
  = x, command = f(), memory = "auto") will use persistent memory in or-
  der to avoid rereading all of x for every branch of y.
- "transient": the target gets unloaded after every new target completes.
  Either way, the target gets automatically loaded into memory whenever an-
  other target needs the value.
- "persistent": the target stays in memory until the end of the pipeline
  (unless storage is "worker", in which case targets unloads the value
  from memory right after storing it in order to avoid sending copious data
  over a network).

For cloud-based file targets (e.g. format = "file" with repository = "aws"),
the memory option applies to the temporary local copy of the file: "persistent"
means it remains until the end of the pipeline and is then deleted, and "transient"
means it gets deleted as soon as possible. The former conserves bandwidth, and
the latter conserves local storage.

garbage_collection

Logical: TRUE to run base::gc() just before the target runs, in whatever R pro-
cess it is about to run (which could be a parallel worker). FALSE to omit garbage

collection. Numeric values get converted to FALSE. The garbage_collection option in `tar_option_set()` is independent of the argument of the same name in `tar_target()`.

deployment   Character of length 1. If deployment is "main", then the target will run on the central controlling R process. Otherwise, if deployment is "worker" and you set up the pipeline with distributed/parallel computing, then the target runs on a parallel worker. For more on distributed/parallel computing in targets, please visit https://books.ropensci.org/targets/crew.html.

priority   Deprecated on 2025-04-08 (targets version 1.10.1.9013). targets has moved to a more efficient scheduling algorithm (https://github.com/ropensci/targets/issues/1458) which cannot support priorities. The priority argument of `tar_target()` no longer has a reliable effect on execution order.

resources   Object returned by tar_resources() with optional settings for high-performance computing functionality, alternative data storage formats, and other optional capabilities of targets. See tar_resources() for details.

storage   Character string to control when the output of the target is saved to storage. Only relevant when using targets with parallel workers (https://books.ropensci.org/targets/crew.html). Must be one of the following values:

- "worker" (default): the worker saves/uploads the value.
- "main": the target's return value is sent back to the host machine and saved/uploaded locally.
- "none": targets makes no attempt to save the result of the target to storage in the location where targets expects it to be. Saving to storage is the responsibility of the user. Use with caution.

retrieval   Character string to control when the current target loads its dependencies into memory before running. (Here, a "dependency" is another target upstream that the current one depends on.) Only relevant when using targets with parallel workers (https://books.ropensci.org/targets/crew.html). Must be one of the following values:

- "auto" (default): equivalent to retrieval = "worker" in almost all cases. But to avoid unnecessary reads from disk, retrieval = "auto" is equivalent to retrieval = "main" for dynamic branches that branch over non-dynamic targets. For example: if your pipeline has tar_target(x, command = f()), then tar_target(y, command = x, pattern = map(x), retrieval = "auto") will use "main" retrieval in order to avoid rereading all of x for every branch of y.
- "worker": the worker loads the target's dependencies.
- "main": the target's dependencies are loaded on the host machine and sent to the worker before the target runs.
- "none": targets makes no attempt to load its dependencies. With retrieval = "none", loading dependencies is the responsibility of the user. Use with caution.

cue   An optional object from tar_cue() to customize the rules that decide whether the target is up to date.

description   Character of length 1, a custom free-form human-readable text description of the
                   target. Descriptions appear as target labels in functions like tar_manifest()
                   and tar_visnetwork(), and they let you select subsets of targets for the names
                   argument of functions like tar_make(). For example, tar_manifest(names =
                   tar_described_as(starts_with("survival model"))) lists all the targets
                   whose descriptions start with the character string "survival model".

Value

A target object. Users should not modify these directly, just feed them to list() in your target
script file (default: _targets.R).

---

use_tarchives   *Use tarchives*

Description

Set up tarchives for an existing package.

Usage

```
use_tarchives(store = targets::tar_config_get("store"))
```

Arguments

store   Character of length 1, path to the targets data store. Defaults to tar_config_get("store"),
                   which in turn defaults to _targets/. When you set this argument, the value
                   of tar_config_get("store") is temporarily changed for the current function
                   call. See tar_config_get() and tar_config_set() for details about how to
                   set the data store path persistently for a project.

Value

No return value, called for side effects.

Index