

Package ‘fledge’

July 22, 2025

Title Smoother Change Tracking and Versioning for R Packages

Version 0.1.2

Date 2024-02-18

Description Streamlines the process of updating changelogs (NEWS.md)
and versioning R packages developed in git repositories.

License GPL-3

URL <https://fledge.cynkra.com/>, <https://github.com/cynkra/fledge>

BugReports <https://github.com/cynkra/fledge/issues>

Imports brio, cli, desc ($\geq 1.2.0$), gert ($\geq 1.4.0$), purrr ($\geq 0.3.2$),
rematch2, rlang ($\geq 0.4.12$), tibble, usethis ($\geq 1.5.0$),
whoami, withr

Suggests mockery, covr, fansi, fs, knitr, parsedate, rmarkdown,
rstudioapi, testthat ($\geq 3.0.0$), callr, pkgload

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.1

Config/testthat/edition 3

NeedsCompilation no

Author Kirill Müller [aut, cre, cph] (ORCID:
<<https://orcid.org/0000-0002-1416-3412>>),
Patrick Schratz [aut] (ORCID: <<https://orcid.org/0000-0003-0748-6624>>),
Maëlle Salmon [ctb] (ORCID: <<https://orcid.org/0000-0002-2815-0399>>)

Maintainer Kirill Müller <kirill@cynkra.com>

Repository CRAN

Date/Publication 2024-02-18 16:30:02 UTC

Contents

bump_version	2
commit_version	3
create_demo_project	4
finalize_version	5
get_last_tag	6
get_top_level_commits	6
tag_version	7
unbump_version	8
update_news	9
update_version	10
with_demo_project	11
Index	12

bump_version	<i>Bump package version</i>
--------------	-----------------------------

Description

- Calls the following functions:
1. Verify that the current branch is the main branch.
 2. [update_news\(\)](#)
 3. [update_version\(\)](#), using the which argument
 4. Depending on the which argument:
 - If "dev", [finalize_version\(\)](#) with push = FALSE
 - Otherwise, [commit_version\(\)](#).

Usage

```
bump_version(which = "dev")
```

Arguments

which	Component of the version number to update. Supported values are "dev" (default), "patch", "minor" and "major".
-------	--

Value

None

Bumped too soon?

Have you just run `bump_version()`, then realized "oh shoot, I forgot to merge that PR"? Fear not, run [unbump_version\(\)](#), merge that PR, run `bump_version()`.

See Also

[unbump_version\(\)](#)

Examples

```
# Create mock package in a temporary directory.
# Set open to TRUE if you want to play in the mock package.
with_demo_project({
  # Use functions as if inside the newly created package project.
  # (Or go and actually run code inside the newly created package project!)
  # Add a new R file.
  usethis::use_r("cool-function", open = FALSE)
  # Pretend we added useful code inside it.
  # Track the new R file with Git.
  gert::git_add("R/cool-function.R")
  gert::git_commit("- Add cool function.")
  # Bump version with fledge.
  fledge::bump_version()
})
```

commit_version

Commits NEWS.md and DESCRIPTION to Git

Description

Commits changes to NEWS.md and DESCRIPTION, amending a previous commit created by **fledge** if necessary.

Usage

```
commit_version()
```

Value

Invisibly: TRUE if a previous commit for that version has been amended, FALSE if not.

Examples

```
# Create mock package in a temporary directory.
# Set open to TRUE if you want to play in the mock package.
with_demo_project({
  # Use functions as if inside the newly created package project.
  # (Or go and actually run code inside the newly created package project!)
  # Add a new R file.
  usethis::use_r("cool-function", open = FALSE)
  # Pretend we added useful code inside it.
  # Track the new R file with Git.
  gert::git_add("R/cool-function.R")
  gert::git_commit("- Add cool function.")
})
```

```
# Bump version with fledge.
fledge::bump_version()
desc::desc_add_author(given = "Jane", family = "Doe", role = "ctb")
fledge::commit_version()
})
```

create_demo_project *Create example repo for fledge demos*

Description

Create example repo for fledge demos

Usage

```
create_demo_project(
  open = rlang::is_interactive(),
  name = "tea",
  maintainer = NULL,
  email = NULL,
  date = "2021-09-27",
  dir = file.path(tempdir(), "fledge"),
  news = FALSE
)
```

Arguments

open	Whether to open the new project.
name	Package name.
maintainer	Name for DESCRIPTION and git.
email	Email for DESCRIPTION and git.
date	String of time for DESCRIPTION and git.
dir	Directory within which to create the mock package folder.
news	If TRUE, create a NEWS.md file.

Value

The path to the newly created mock package.

finalize_version	<i>Finalize package version</i>
------------------	---------------------------------

Description

Calls the following functions:

1. `commit_version()`
2. `tag_version()`, setting `force = TRUE` if and only if `commit_version()` amended a commit.
3. Force-pushes the created tag to the "origin" remote, if `push = TRUE`.

Usage

```
finalize_version(push = FALSE)
```

Arguments

push	If TRUE, push the created tag.
------	--------------------------------

Value

None

Examples

```
# Create mock package in a temporary directory.
# Set open to TRUE if you want to play in the mock package.
with_demo_project({
  # Use functions as if inside the newly created package project.
  # (Or go and actually run code inside the newly created package project!)
  # Add a new R file.
  usethis::use_r("cool-function", open = FALSE)
  # Pretend we added useful code inside it.
  # Track the new R file with Git.
  gert::git_add("R/cool-function.R")
  gert::git_commit("- Add cool function.")
  # Bump version with fledge.
  fledge::bump_version()
  # Edit news by hand
  # ...
  # Once done
  fledge::finalize_version()
})
```

get_last_tag	<i>The most recent tag</i>
--------------	----------------------------

Description

Returns the most recent Git tag.

Usage

```
get_last_tag()
```

Value

A one-row tibble with columns name, ref and commit. For annotated tags (as created by fledge), commit may be different from the SHA of the commit that this tag points to. Use `gert::git_log()` to find the actual commit.

Examples

```
# Create mock package in a temporary directory.
# Set open to TRUE if you want to play in the mock package.
with_demo_project({
  # Use functions as if inside the newly created package project.
  # (Or go and actually run code inside the newly created package project!)
  # Add a new R file.
  usethis::use_r("cool-function", open = FALSE)
  # Pretend we added useful code inside it.
  # Track the new R file with Git.
  gert::git_add("R/cool-function.R")
  gert::git_commit("- Add cool function.")
  # Bump version with fledge.
  fledge::bump_version()
  fledge::finalize_version()
  print(get_top_level_commits(since = NULL))
  print(fledge::get_last_tag())
})
```

get_top_level_commits	<i>All top-level commits</i>
-----------------------	------------------------------

Description

Return all top-level commits since a particular version as commit objects.

Usage

```
get_top_level_commits(since = NULL)
```

Arguments

since A commit SHA, e.g. as returned from `get_last_tag()`. If NULL, the entire log is retrieved.

Value

A [tibble](#) with at least two columns:

- commit: the commit SHA
- message: the commit message

Examples

```
# Create mock package in a temporary directory.
# Set open to TRUE if you want to play in the mock package.
with_demo_project({
  # Use functions as if inside the newly created package project.
  # (Or go and actually run code inside the newly created package project!)
  # Add a new R file.
  usethis::use_r("cool-function", open = FALSE)
  # Pretend we added useful code inside it.
  # Track the new R file with Git.
  gert::git_add("R/cool-function.R")
  gert::git_commit("- Add cool function.")
  # Bump version with fledge.
  fledge::bump_version()
  fledge::finalize_version()
  print(get_top_level_commits(since = NULL))
  print(fledge::get_last_tag())
})
```

tag_version	<i>Create a new version tag</i>
-------------	---------------------------------

Description

Parses NEWS.md and creates/updates the tag for the most recent version.

Usage

```
tag_version(force = FALSE)
```

Arguments

force Re-tag even if the last commit wasn't created by `bump_version()`. Useful when defining a CRAN release.

Value

The created tag, invisibly.

None

Examples

```
# Create mock package in a temporary directory.
# Set open to TRUE if you want to play in the mock package.
with_demo_project({
  # Use functions as if inside the newly created package project.
  # (Or go and actually run code inside the newly created package project!)
  # Add a new R file.
  usethis::use_r("cool-function", open = FALSE)
  # Pretend we added useful code inside it.
  # Track the new R file with Git.
  gert::git_add("R/cool-function.R")
  gert::git_commit("- Add cool function.")
  # Bump version with fledge.
  fledge::bump_version()
  fledge::update_news(c("- something I forgot", "- blabla"))
  fledge::update_version()
  gert::git_add("NEWS.md")
  gert::git_commit(message = "release notes tweaking")
  fledge::tag_version()
  print(fledge::get_last_tag())
})
```

unbump_version

Undoes bumping the package version

Description

This undoes the effect of a [bump_version\(\)](#) call, with a safety check.

Usage

```
unbump_version()
```

Value

NULL, invisibly. This function is called for its side effects.

None

See Also

[bump_version](#)

Examples

```
# Create mock package in a temporary directory.
# Set open to TRUE if you want to play in the mock package.
with_demo_project({
  # Use functions as if inside the newly created package project.
  # (Or go and actually run code inside the newly created package project!)
  # Add a new R file.
  usethis::use_r("cool-function", open = FALSE)
  # Pretend we added useful code inside it.
  # Track the new R file with Git.
  gert::git_add("R/cool-function.R")
  gert::git_commit("- Add cool function.")
  # Bump version with fledge.
  fledge::bump_version()
  # Oh no, we forgot to also add the awesome function for that version!
  # UNBUMP
  fledge::unbump_version()
  # Add a new R file.
  usethis::use_r("awesome-function", open = FALSE)
  # Pretend we added awesome code inside it.
  # Track the new R file with Git.
  gert::git_add("R/awesome-function.R")
  gert::git_commit("- Add awesome function.")
  # Bump version with fledge.
  fledge::bump_version()
  #'
})
```

update_news

Update NEWS.md with messages from top-level commits

Description

Lists all commits from a range (default: top-level commits since the most recent tag) and adds bullets from their body to NEWS.md. Creates NEWS.md if necessary.

Usage

```
update_news(messages = NULL)
```

Arguments

messages	A character vector of commit messages, e.g. as in the message column in the return value of get_top_level_commits() . The default uses the top level commits since the last tag as retrieved by get_last_tag() .
----------	--

Value

None

Examples

```
# Create mock package in a temporary directory.
# Set open to TRUE if you want to play in the mock package.
with_demo_project({
  # Use functions as if inside the newly created package project.
  # (Or go and actually run code inside the newly created package project!)
  # Add a new R file.
  usethis::use_r("cool-function", open = FALSE)
  # Pretend we added useful code inside it.
  # Track the new R file with Git.
  gert::git_add("R/cool-function.R")
  gert::git_commit("- Add cool function.")
  # Bump version with fledge.
  fledge::bump_version()
  fledge::update_news(c("- something I forgot", "- blabla"))
  fledge::update_version()
  gert::git_add("NEWS.md")
  gert::git_commit(message = "release notes tweaking")
  fledge::tag_version()
  print(fledge::get_last_tag())
})
```

update_version

Update NEWS.md and DESCRIPTION with a new version

Description

Bumps a version component and adds to NEWS.md and DESCRIPTION.

Usage

```
update_version(which = "dev")
```

Arguments

which	Component of the version number to update. Supported values are "dev" (default), "patch", "minor" and "major".
-------	--

Value

None

Examples

```
# Create mock package in a temporary directory.
# Set open to TRUE if you want to play in the mock package.
with_demo_project({
  # Use functions as if inside the newly created package project.
  # (Or go and actually run code inside the newly created package project!)
```

```

# Add a new R file.
usethis::use_r("cool-function", open = FALSE)
# Pretend we added useful code inside it.
# Track the new R file with Git.
gert::git_add("R/cool-function.R")
gert::git_commit("- Add cool function.")
# Bump version with fledge.
fledge::bump_version()
fledge::update_news(c("- something I forgot", "- blabla"))
fledge::update_version()
gert::git_add("NEWS.md")
gert::git_commit(message = "release notes tweaking")
fledge::tag_version()
print(fledge::get_last_tag())
})

```

with_demo_project	<i>Run code in temporary project</i>
-------------------	--------------------------------------

Description

Run code in temporary project

Usage

```
with_demo_project(code, dir = NULL, news = TRUE, quiet = FALSE)
```

Arguments

code	Code to run with temporary active project
dir	Directory within which to create the mock package folder.
news	If TRUE, create a NEWS.md file.
quiet	Whether to show messages from usethis

Value

None

Examples

```

with_demo_project({
  # Add a new R file.
  usethis::use_r("cool-function", open = FALSE)
  # Pretend we added useful code inside it.
  # Track the new R file with Git.
  gert::git_add("R/cool-function.R")
  gert::git_commit("- Add cool function.")
  # Bump version with fledge.
  fledge::bump_version()
})

```

Index

bump_version, [2](#)
bump_version(), [7](#), [8](#)
bump_version_impl (bump_version), [2](#)

commit_version, [3](#)
commit_version(), [2](#), [5](#)
create_demo_project, [4](#)

finalize_version, [5](#)
finalize_version(), [2](#)
finalize_version_impl
 (finalize_version), [5](#)

gert::git_log(), [6](#)
get_last_tag, [6](#)
get_last_tag(), [7](#), [9](#)
get_top_level_commits, [6](#)
get_top_level_commits(), [9](#)

tag_version, [7](#)
tag_version(), [5](#)
tibble, [7](#)

unbump_version, [8](#)
unbump_version(), [2](#), [3](#)
unbump_version_impl (unbump_version), [8](#)
update_news, [9](#)
update_news(), [2](#)
update_version, [10](#)
update_version(), [2](#)

with_demo_project, [11](#)