Package 'cffr'

January 30, 2023

Title Generate Citation File Format ('cff') Metadata for R Packages

Version 0.4.1

Description The Citation File Format version 1.2.0 <doi:10.5281/zenodo.5171937> is a human and machine readable file format which provides citation metadata for software. This package provides core utilities to generate and validate this metadata.

License GPL (>= 3)

URL https://docs.ropensci.org/cffr/, https://github.com/ropensci/cffr

BugReports https://github.com/ropensci/cffr/issues

Depends R (>= 3.6.0)

Imports cli (>= 2.0.0), desc (>= 1.3.0), jsonlite (>= 1.7.2), jsonvalidate (>= 1.1.0), yaml (>= 2.2.1)

Suggests bibtex (>= 0.5.0), knitr, lifecycle, rmarkdown, testthat (>= 3.0.0), usethis

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

X-schema.org-isPartOf https://ropensci.org

X-schema.org-keywords attribution, citation, credit, citation-files, cff, metadata

NeedsCompilation no

Author Diego Hernangómez [aut, cre, cph] (<https://orcid.org/0000-0001-8457-4658>), João Martins [rev] (<https://orcid.org/0000-0001-7961-4280>), Scott Chamberlain [rev] (<https://orcid.org/0000-0003-1444-9135>)

Maintainer Diego Hernangómez <diego.hernangomezherrero@gmail.com>

Repository CRAN

Date/Publication 2023-01-30 22:00:02 UTC

R topics documented:

`cff_create` *Create* `cff` *object*

Description

Create a `cff` object from a given source for further manipulation. Similar to `cff_write()`, but returns a object rather than writing directly to a file. See **Examples**.

Usage

```
cff_create(
  x,
  keys = list(),
  cff_version = "1.2.0",
  gh_keywords = TRUE,
  dependencies = TRUE
)
```

Arguments

`x`	The source that would be used for generating the `cff` object. It could be: • A missing value. That would retrieve the DESCRIPTION file on your in-development package. • An existing `cff` object, • The name of an installed package (`"jsonlite"`), or • Path to a DESCRIPTION file (`"*/DESCRIPTION*"`).
`keys`	List of additional keys to add to the `cff` object. See **Details**.

`cff_version`	The Citation File Format schema version that the `CITATION.cff` file adheres to for providing the citation metadata.
`gh_keywords`	Logical `TRUE/FALSE`. If the package is hosted on GitHub, would you like to add the repo topics as keywords?
`dependencies`	Logical `TRUE/FALSE`. Would you like to add the of your package to the `reference` key?

Details

It is possible to add additional keys not detected by `cff_create()` using the `keys` argument. A list of valid keys can be retrieved with `cff_schema_keys()`.

Please refer to Guide to Citation File Format schema version 1.2.0. for additional details.

If `x` is a path to a DESCRIPTION file or `inst/CITATION`, is not present on your package, **cffr** would auto-generate a `preferred-citation` key using the information provided on that file. On

Value

A `cff` list object.

See Also

Guide to Citation File Format schema version 1.2.0.

`vignette("cffr", "cffr")`

Other core functions: `cff_read()`, `cff_validate()`, `cff_write()`

Examples

```
# Installed package
cff_create("jsonlite")

# Demo file
demo_file <- system.file("examples/DESCRIPTION_basic", package = "cffr")
cff_create(demo_file)

# Add additional keys

newkeys <- list(
  message = "This overwrites fields",
  abstract = "New abstract",
  keywords = c("A", "new", "list", "of", "keywords"),
  authors = list(cff_parse_person("New author"))
)

cff_create(demo_file, keys = newkeys)

# Update a field on a list - i,e: authors, contacts, etc.
# We are adding a new contact here
```

```
old <- cff_create(demo_file)

new_contact <- append(
  old$contact,
  list(
    cff_parse_person(person(
      given = "I am",
      family = "New Contact"
    ))
  )
)


cff_create(demo_file, keys = list("contact" = new_contact))
```

`cff_extract_to_bibtex` *Create BibTeX entries from a package*

Description

Extract the information of a package to BibTeX. This is done by creating a `cff` object with `cff_create()` and extracting the corresponding entries with `cff_to_bibtex()`.

Usage

```
cff_extract_to_bibtex(x, what = "preferred")
```

Arguments

`x`	The source that would be used for generating the `cff` object. It could be: • A missing value. That would retrieve the DESCRIPTION file on your in-development package. • An existing `cff` object, • The name of an installed package (`"jsonlite"`), or • Path to a DESCRIPTION file (`"*/DESCRIPTION*"`).
`what`	Fields to extract. The value could be: • `preferred`: This would create a single entry with the main citation info of the package. • `references`: Extract all the entries on `references`. • `all`: A combination of the previous two options. This would extract both the preferred citation info and the references.

Value

A `bibentry` object or a list of `bibentry` objects. This could be parsed to BibTeX using `toBibtex()`

See Also

Other bibtex: cff_from_bibtex(), cff_to_bibtex(), encoded_utf_to_latex(), write_bib()

Examples

```
jsonvalidate <- cff_extract_to_bibtex("jsonvalidate")

jsonvalidate

toBibtex(jsonvalidate)

lite <- cff_extract_to_bibtex("jsonlite", "references")

lite

toBibtex(lite)
```

---

cff_from_bibtex *Create a cff object from BibTeX entries*

Description

Extract the information of a BibTeX file or BibTeX entry and creates the corresponding cff object with cff_parse_citation().

Usage

```
cff_from_bibtex(x, encoding = "UTF-8", ...)
```

Arguments

x	The source that would be used for generating the cff object. A character object indicating either: • The path to a BibTeX file. • A vector of characters with the full BibTeX string. See **Examples**
encoding	Encoding to be assumed for x. See readLines().
...	Other arguments passed to bibtex::read.bib().

Details

This function requires the package **bibtex** (>= 0.5.0), that is listed as Suggested by **cffr**.

Value

A cff object ready to be used on cff_create().

See Also

vignette("bibtex_cff", package = "cffr") to learn about the mapping of information between BibTeX and CITATION.cff.

Other bibtex: cff_extract_to_bibtex(), cff_to_bibtex(), encoded_utf_to_latex(), write_bib()

Examples

```
if (requireNamespace("bibtex", quietly = TRUE)) {
  x <- c(
    "@book{einstein1921,
    title        = {Relativity: The Special and the General Theory},
    author       = {Einstein, Albert},
    year         = 1920,
    publisher    = {Henry Holt and Company},
    address      = {London, United Kingdom},
    isbn         = 9781587340925
}",
    "@misc{misc-full,
    title        = {Handing out random pamphlets in airports},
    author       = {Joe-Bob Missilany},
    year         = 1984,
    month        = oct,
    note         = {This is a full MISC entry},
    howpublished = {Handed out at O'Hare}
}"
  )

  cff_from_bibtex(x)

  # From a file

  x2 <- system.file("examples/example.bib", package = "cffr")
  cff_from_bibtex(x2)
}
```

---

cff_gha_update *Install a cffr GitHub Action*

Description

This function would install a GitHub Action on your repo. The action will update your CITATION.cff when any of these events occur:

- You publish a new release of the package.
- Your DESCRIPTION or inst/CITATION are modified.
- The action can be run also manually.

Usage

```
cff_gha_update(path = ".", overwrite = FALSE)
```

Arguments

path	Project directory
overwrite	If already present, do you want to overwrite your action?

Details

Triggers on your action can be modified, see Events that trigger workflows.

Value

Invisible, this function is called by its side effects.

See Also

Other git: cff_git_hook

Examples

```
## Not run:
cff_gha_update()

## End(Not run)
```

---

cff_git_hook *Use a git pre-commit hook* **[Experimental]**

Description

Install a pre-commit hook that remembers you to update your CITATION.cff file.

Usage

```
cff_git_hook_install()

cff_git_hook_remove()
```

Details

This function would install a pre-commit hook using usethis::use_git_hook().

A pre-commit hook is a script that identifies simple issues before submission to code review. This pre-commit hook would warn you if any of the following conditions are met:

- You included in a commit your DESCRIPTION or inst/CITATION file, you are not including your CITATION.cff and the CITATION.cff file is "older" than any of your DESCRIPTION or inst/CITATION file, or
- You have updated your CITATION.cff but you are not including it on your commit.

Value

Invisible. This function is called for its side effects.

A word of caution

The pre-commit hook may prevent you to commit if you are not updating your `CITATION.cff`. However, the mechanism of detection is not perfect and would be triggered also even if you have tried to update your `CITATION.cff` file.

This is typically the case when you have updated your DESCRIPTION or inst/CITATION files but those changes doesn't make a change on your `CITATION.cff` file (i.e. you are including new dependencies).

In those cases, you can override the check running `git commit --no-verify` on the Terminal tab. If you are using RStudio you can run also this command from a R script by selecting that line and sending it to the Terminal using:

- `Ctrl+Alt+Enter` (Windows & Linux), or
- `Cmd+Option+Return` (Mac).

Removing the git pre-commit hook

You can remove the pre-commit hook by running `cff_git_hook_remove()`.

See Also

`usethis::use_git_hook()`, `usethis::use_git()`

Other git: `cff_gha_update()`

Examples

```
## Not run:
cff_git_hook_install()

## End(Not run)
```

---

cff_parse_citation *Parse a* bibentry *to* cff

Description

Parse a `bibentry` object to a valid format for a `CITATION.cff` file.

Usage

```
cff_parse_citation(bib)
```

Arguments

bib	A bibentry object, either created with bibentry() (preferred) or citEntry().

Details

This is a helper function designed to help on adding or replacing the auto-generated authors of the package. See **Examples**.

This function tries to adapt a bibentry object (generated with bibentry() or citEntry()) to the CFF standard.

Entry types considered:

- **Article**, **Book**, **Booklet**, **InBook**, **InCollection**, **InProceedings**, **Manual**, **MastersThesis**, **Misc**, **PhDThesis**, **Proceedings**, **TechReport**, **Unpublished**. See bibentry() for more information.

Note that **Conference** is not implemented in bibentry(), however is equivalent to **InProceedings** (Patashnik (1988)).

Fields considered:

- **address**, **author**, **booktitle**, **chapter**, **edition**, **editor**, **howpublished**, **institution**, **journal**, **key**, **month**, **note**, **number**, **organization**, **pages**, **publisher**, **school**, **series**, **title**, **type**, **year**.

annote and **crossref** fields are ignored.

Value

A cff object ready to be used on cff_create().

References

- Patashnik, Oren. "BIBTEXTING" February 1988. https://osl.ugr.es/CTAN/biblio/bibtex/base/btxdoc.pdf.
- Haines, R., & The Ruby Citation File Format Developers. (2021). *Ruby CFF Library (Version 0.9.0)* (Computer software). doi:10.5281/zenodo.1184077.

See Also

cff_create(), vignette("bibtex_cff", "cffr"), bibentry()

Other parsers: cff_parse_person()

Examples

```
bib <- citation("base")
bib


# To cff
bib_to_cff <- cff_parse_citation(bib)
```

```
bib_to_cff

# Create the object
new_cff <- cff()

full <- cff_create(new_cff, keys = list("preferred-citation" = bib_to_cff))

full
# Validate
cff_validate(full)

# Several citations

cff_parse_citation(citation("rmarkdown"))
```

`cff_parse_person` *Parse a person to* `cff`

Description

Parse a person or string to a valid format for a `CITATION.cff` file. This is a helper function designed to help on adding or replacing the auto-generated authors of the package.

Usage

```
cff_parse_person(person)

cff_parse_person_bibtex(person)
```

Arguments

`person`	A `person` object created with `person()` or a character string. See **Details**.

Details

The `person` parameter of the function could be:

- For `cff_parse_person()`: A `person` object or a character coercible to `person`. See `person()` for details.
- For `cff_parse_person_bibtex()`: A string with the definition of an author or several authors, using the standard BibTeX notation. See Markey (2007) for a full explanation.

See **Examples** for more information.

Value

A `cff` object ready to be used on `cff_create()`.

References

- Patashnik, Oren. "BIBTEXTING" February 1988. https://osl.ugr.es/CTAN/biblio/bibtex/base/btxdoc.pdf.
- Markey, Nicolas. "Tame the BeaST." *The B to X of BibTeX, Version 1.4* (October 2007). https://osl.ugr.es/CTAN/info/bibtex/tamethebeast/ttb_en.pdf.

See Also

cff_create(), vignette("cffr", "cffr"), utils::person()

Other parsers: cff_parse_citation()

Examples

```
# Parse a person object

cff_parse_person(person(
  given = "First",
  family = "Author",
  role = c("aut", "cre"),
  email = "first.last@example.com",
  comment = c(
    ORCID = "0000-0001-8457-4658",
    affiliation = "An affiliation"
  )
))

# Parse a string

cff_parse_person("Julio Iglesias <fake@email.com>")

# Several persons
persons <- c(person("Clark", "Kent"), person("Lois", "Lane"))

cff_parse_person(persons)

# Or you can use BibTeX style if you prefer

x <- "Frank Sinatra and Dean Martin and Davis, Jr., Sammy and Joey Bishop"

cff_parse_person_bibtex(x)

cff_parse_person_bibtex("Herbert von Karajan")
```

---

cff_read *Read and manipulate* cff *objects*

Description

A class and utility methods for reading, creating and holding CFF information.

Usage

```
cff_read(path)

cff(path, ...)

as.cff(x)
```

Arguments

path	The path to a CITATION.cff file.
...	Named arguments to be used for creating a cff object. See **Details**.
x	a character string for the as.cff default method

Details

This object can be manipulated using cff_create().

Note that this function reads CITATION.cff files. If you want to create cff objects from DESCRIPTION files use cff_create().

If no additional ... parameters are supplied (the default behavior), a minimal valid cff object is created. Valid parameters are those specified on cff_schema_keys():

valid cff keys
cff-version
message
type
license
title
version
doi
abstract
authors
preferred-citation
repository
repository-artifact
repository-code
url
date-released
contact
keywords
references
commit
identifiers
license-url

Value

A `cff` object. Under the hood, a `cff` object is a regular `list` object with a special `print()` method.

See Also

Other core functions: `cff_create()`, `cff_validate()`, `cff_write()`

Examples

```
# Blank cff
cff()

# From file
cff_read(system.file("examples/CITATION_basic.cff",
  package = "cffr"
))

# Use custom params
test <- cff(
  title = "Manipulating files",
  keywords = c("A", "new", "list", "of", "keywords"),
  authors = list(cff_parse_person("New author"))
)
test

# Would fail
cff_validate(test)


# Modify with cff_create
new <- cff_create(test, keys = list(
  "cff-version" = "1.2.0",
  message = "A blank file"
))
new

# Would pass
cff_validate(new)



# Convert a list to "cff" object
cffobj <- as.cff(list(
  "cff-version" = "1.2.0",
  title = "Manipulating files"
))

class(cffobj)

# Nice display thanks to yaml package
cffobj
```

`cff_schema` *Schema utils*

Description

Helper functions with the valid values of different fields, according to the Citation File Format schema version 1.2.0.

- `cff_schema_keys()` provides the valid high-level keys of the Citation File Format.
- `cff_schema_keys_license()` provides the valid SPDX license identifier(s) to be used on the `CITATION.cff` file.
- `cff_schema_definitions_person()` and `cff_schema_definitions_entity()` returns the valid fields to be included when defining a person or entity.
- `cff_schema_definitions_refs()` provides the valid keys to be used on the `preferred-citation` and `references` keys.

Usage

```
cff_schema_keys(sorted = FALSE)

cff_schema_keys_license()

cff_schema_definitions_person()

cff_schema_definitions_entity()

cff_schema_definitions_refs()
```

Arguments

`sorted`	Logical `TRUE/FALSE`. Should the keys be arranged alphabetically?

Value

A vector of characters with the names of the valid keys to be used on a Citation File Format version 1.2.0

Source

Guide to Citation File Format schema version 1.2.0.

Examples

```
cff_schema_keys(sorted = TRUE)

# Valid Licenses keys
```

```
head(cff_schema_keys_license(), 20)

cff_schema_definitions_person()

cff_schema_definitions_entity()

cff_schema_definitions_refs()
```

`cff_to_bibtex` *Create a BibTeX entry from a CITATION.cff file or a* `cff` *object*

Description

Creates a `bibentry` object (`bibentry()`) from a `cff` object

Usage

```
cff_to_bibtex(x)
```

Arguments

`x`	The source that would be used for generating the `cff` object. It could be: • An existing `cff` object, • A CITATION.cff file.

Value

A `bibentry` object that can be parsed to BibTeX format with `toBibtex()`

References

- Patashnik, Oren. "BIBTEXTING" February 1988. https://osl.ugr.es/CTAN/biblio/bibtex/base/btxdoc.pdf.
- Haines, R., & The Ruby Citation File Format Developers. (2021). *Ruby CFF Library (Version 0.9.0)* (Computer software). doi:10.5281/zenodo.1184077.

See Also

`cff_parse_citation()`, `bibentry()`, `toBibtex()`

Other bibtex: `cff_extract_to_bibtex()`, `cff_from_bibtex()`, `encoded_utf_to_latex()`, `write_bib()`

Examples

```
# From a cff object
package <- cff_create("rmarkdown")

obj <- cff_to_bibtex(package)

obj

toBibtex(obj)

# Same info as
toBibtex(citation("rmarkdown")[1])
```

`cff_validate` *Validate a* `CITATION.cff` *file or a* `cff` *object*

Description

Validate a `CITATION.cff` file or a `cff` object created with `cff_create()` using the corresponding validation schema.json.

Usage

```
cff_validate(x = "CITATION.cff", verbose = TRUE)
```

Arguments

`x`	This is expected to be either a `cff` object created with `cff_create()` or the path to a `CITATION.cff` file to be validated.
`verbose`	Logical `TRUE/FALSE`. On `TRUE` the function would display informative messages.

Value

A message indicating the result of the validation and an invisible value `TRUE/FALSE`.

See Also

Guide to Citation File Format schema version 1.2.0.

Other core functions: `cff_create()`, `cff_read()`, `cff_write()`

Examples

```
# Full .cff example
cff_validate(system.file("examples/CITATION_complete.cff", package = "cffr"))

# Validate a cffr object
cffr <- cff_create("jsonlite")
class(cffr)
cff_validate(cffr)

## Not run:
# .cff with errors
cff_validate(system.file("examples/CITATION_error.cff", package = "cffr"))
# If a CITATION file (note that is not .cff) it throws an error
cff_validate(system.file("CITATION", package = "cffr"))

## End(Not run)
```

`cff_write` *Write a* `CITATION.cff` *file*

Description

This is the core function of the package and likely to be the only one you would need when developing a package.

This function writes out a `CITATION.cff` file for a given package. This function is basically a wrapper around `cff_create()` to both create the `cff` object and writes it out to a YAML-formatted file in one command.

Usage

```
cff_write(
  x,
  outfile = "CITATION.cff",
  keys = list(),
  cff_version = "1.2.0",
  gh_keywords = TRUE,
  dependencies = TRUE,
  validate = TRUE,
  verbose = TRUE
)
```

Arguments

`x`	The source that would be used for generating the `CITATION.cff` file. It could be: • A missing value. That would retrieve the DESCRIPTION file on your in-development package.

- A cff object,
- The name of an installed package ("jsonlite"), or
- Path to a DESCRIPTION file ("*/DESCRIPTION*").

outfile	The name and path of the CITATION.cff to be created.
keys	List of additional keys to add to the cff object. See cff_create() for details and examples.
cff_version	The Citation File Format schema version that the CITATION.cff file adheres to for providing the citation metadata.
gh_keywords	Logical TRUE/FALSE. If the package is hosted on GitHub, would you like to add the repo topics as keywords?
dependencies	Logical TRUE/FALSE. Would you like to add the of your package to the reference key?
validate	Logical TRUE/FALSE. Should the new file be validated using cff_validate()?
verbose	Logical TRUE/FALSE. On TRUE the function would display informative messages.

Details

When creating and writing a CITATION.cff for the first time, the function adds "CITATION.cff" to ".Rbuildignore".

Value

A CITATION.cff file and an (invisible) cff object.

See Also

Guide to Citation File Format schema version 1.2.0.

Other core functions: cff_create(), cff_read(), cff_validate()

Examples

```
tmpfile <- tempfile(fileext = ".cff")
cff_obj <- cff_write("jsonlite", outfile = tmpfile)

cff_obj

# Force clean-up
file.remove(tmpfile)
```

cran_to_spdx *Mapping between* License *fields and SPDX*

Description

A dataset containing the mapping between the License strings observed on CRAN packages and its (approximate) match on the SPDX License List.

Usage

```
cran_to_spdx
```

Format

A data frame with 91 rows and 2 variables:

- LICENSE: A valid License string on CRAN.
- SPDX. A valid SPDX License Identifier.

Source

https://spdx.org/licenses/

See Also

Writing R Extensions, Licensing section.

Examples

```
data("cran_to_spdx")

head(cran_to_spdx, 20)
```

write_bib *Create a .bib file*

Description

Creates a .bib file from a bibentry object(s)

Usage

```
write_bib(x, file = NULL, append = FALSE, verbose = TRUE, ascii = FALSE)
```

Arguments

x	A bibentry object created with: • cff_extract_to_bibtex(), cff_to_bibtex() • citation() or bibentry()
file	Name of the file. If NULL it would display the lines to be written.
append	Whether to append the entries to an existing file or not.
verbose	Display informative messages
ascii	Whether to write the entries using ASCII characters only or not.

Details

For security reasons, if the file already exists the function would create a backup copy on the same directory.

Value

Writes an .bib file specified on file parameter and the equivalent Bibtex object created with utils::toBibtex().

See Also

vignette("bibtex_cff", "cffr"), knitr::write_bib() and the following packages:

- bibtex package.
- RefManageR package.
- rbibutils

Other bibtex: cff_extract_to_bibtex(), cff_from_bibtex(), cff_to_bibtex(), encoded_utf_to_latex()

Examples

```
bib <- bibentry("Misc",
  title = "My title",
  author = "Fran Pérez"
)

write_bib(bib)

write_bib(bib, ascii = TRUE)
```

Index