

Citation File Format (CFF)

Stephan Druskat (mail@sdruskat.net)

06 October 2017

Abstract

The *Citation File Format (CFF)* is a human- and machine-readable format for CITATION files. These files provide citation metadata for (research and scientific) software. The format aims to support all use cases for software citation described in [1]. CFF is serialized in YAML 1.2, and is therefore Unicode-based and cross-language (in terms of both natural language scripts and programming languages). This specification, together with the Unicode standard for characters, aims to provide all the information necessary to understand CFF, and to use (i.e., write) and re-use (i.e., read, validate, convert from) it. These specifications are maintained openly at <https://github.com/sdruskat/citation-file-format>.

Contents

| | |
|---|-----------|
| Introduction | 2 |
| Status of this document | 2 |
| Rationale | 2 |
| Goals | 3 |
| Concepts | 3 |
| Format | 3 |
| File structure | 3 |
| Reference structure | 3 |
| Notable reference keys | 4 |
| Formatting | 5 |
| Reference keys | 5 |
| Exemplary uses | 7 |
| Reference types | 9 |
| Objects | 10 |
| Entity objects | 10 |
| Exemplary uses | 10 |
| Person objects | 11 |
| Exemplary uses | 11 |
| Person roles | 12 |
| Specified value strings | 12 |
| Status strings | 12 |
| Language strings | 13 |
| Programming language strings | 13 |
| Schema | 21 |
| Examples | 22 |
| Software examples | 22 |
| A software with a DOI | 22 |
| A software without a DOI | 23 |
| software (with two references) | 23 |

| | |
|---|-----------|
| software-code (without a DOI: code repository + commit) | 24 |
| software-container | 24 |
| software-executable | 25 |
| Other examples | 25 |
| art | 25 |
| article | 25 |
| blog | 26 |
| book | 26 |
| conference-paper | 26 |
| edited-work | 27 |
| report | 27 |
| thesis | 28 |
| Infrastructure | 28 |
| Contributions | 28 |
| License | 28 |
| References | 28 |

Introduction

Status of this document

This document reflects the version 0.9-RC1 of the *Citation File Format* (CFF). CFF has been developed in the context of the *Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1)*, which was held on 6 September 2017 in Manchester, UK. More specifically, the constraints for CFF has been developed in the discussion and speed blogging group “Development and implementation of a standard format for CITATION files”, whose members were Stephan Druskat (Humboldt-Universität zu Berlin, Germany), Neil Chue Hong (Software Sustainability Institute, University of Edinburgh, UK), Raniere Silva (Software Sustainability Institute, University of Manchester, UK), Radovan Bast (University of Tromsø, Norway), Andrew Rowley (University of Manchester, UK), and Alexander Konovalov (University of St. Andrews, UK).

CFF Version 0.9-RC1 has been developed by Stephan Druskat with contributions from the following.

- Radovan Bast (@bast): Reporter
- Raniere Silva (@rgaiacs): Reporter

CFF has been developed to provide the first iteration of a format for CITATION files which could be recommended to readers of the blog post which has been produced by the group during the workshop and shortly after, and which will be published on the blog page of the Software Sustainability Institute.

Rationale

The rationale for a standardized, machine- and human-readable format for CITATION files is discussed in more detail in [2]. CFF has been developed to support all use cases for the citation of software, as discussed in [1], and thus promote attribution and credit for software in general, and research software in particular.

In a blog post [3], Robin Wilson has introduced CITATION files as a means to make citation information for software easily accessible. This accessibility is important, because in order to receive deserved credit for research software in the academic system - where credit is still mainly measured based on citations -, the citation information for software must be made visible; Authors will only cite software if the citation information is readily available, as there is no standard, easily deducible way (yet) to cite software, such as there is for journals for example.

Some have followed the advice, and have uploaded CITATION (or CITATION.md, or CITATION.txt) files to the root of the source code repository holding their software. While this practice has made for a good start, plain text, unstandardized CITATION files are not machine-readable, and machine- readability is a precondition for re-use of

the citation information in different contexts which could further support a fair distribution of credit for research software.

Goals

The goal of CFF is to provide an all-purpose citation format (similar to BibTeX or RIS), and specifically provide optimized means of citation for software via the provision of software-specific reference keys and types, e.g., a dedicated type for source code and one for executables, and a reference key for versions, cf. Reference types.

The ultimate goal of CFF as a project is comprehensive uptake and re-use of the format by Research Software Engineers and software developers as well as by vendors and services, such as software repositories, reference managers, etc., in order to boost the visibility of citation information for research software, and empower the fair distribution of credit for software development, maintenance, etc., in academia.

Concepts

For users of other reference formats, such as BibTeX or RIS, it is important to note that in CFF, all available keys can be used for all reference types. CFF leaves reasonability of use with format users and providers of tooling, such as conversion software for CFF and other formats. In other words, the use of keys should follow common sense. If not, it will confuse the user of the CITATION file, and some of the information will probably be lost in re-use scenarios such as conversion or display. If you feel that CFF does not offer a solution for your specific use case, please consider contributing to the format as described in section Contributions.

Furthermore please note that if a section of a work is referenced, this is not supported by a dedicated reference type. Instead, the `section` key in the parent type (i.e., `book` for a section of a book, etc.) should be used.

Format

CFF CITATION files must be named `CITATION.cff`.

CFF is implemented in YAML 1.2, as the language provides optimal human- readability and the required core data types. For details, see the YAML 1.2 Specifications [4].

File structure

CFF CITATION files are YAML 1.2 dictionaries (“maps”) with three mandatory keys: `cff-version`, `message`, `references`.

`cff-version` must specify the exact version of the Citation File Format that is used for the file.

`message` must specify instructions to users on how to cite the software the CITATION.cff file is associated with.

`references` must specify a list of references.

Example:

```
cff-version: 1.0.0
message: "Please cite the following works when using this software."
references:
  - ...
  - ...
```

Reference structure

A reference item, i.e., an item in the list under `references`, must at least specify values for the following mandatory keys: `type`, `authors`, `title`.

`type` must specify the type of the referenced work. For a list of available values, cf. reference types.

`authors` must specify a list of person objects.

`title` must specify the title of the referenced work.

Additionally, it can contain any further reference keys. In version 0.9-RC1, CFF does not specify a strict schema where specific reference types can only contain specific reference keys, although this may be implemented in future versions.

Notable reference keys

`conference`, `database-provider`, `institution`, `publisher`

These keys take an entity object as value. Entity objects reference named entities and provide a fixed set of keys, such as `name` and contact information.

Example:

```
references:
- type: book
  publisher:
  - name: PeerJ
    city: London
    country: GB
    website: https://peerj.com/
```

`authors`, `contact`, `editors`, `editors-series`, `recipients`, `senders`, `translators`

These keys take a collection of person objects as value. Person objects provide a fixed set of keys to reference individuals, including a detailed set for specifying personal names, an affiliation, a role, etc.

Example:

```
references:
- type: software
  authors:
  - family-names: Druskat
    given-names: Stephan
    orcid: 0000-0003-4925-7248
    affiliation: "Humboldt-Universität zu Berlin"
    email: "mail@sdruskat.net"
    website: http://sdruskat.net
    role: main-author
  - family-names: Beethoven
    name-particle: van
    given-names: Ludwig
    role: artist
  - family-names: Fernández de Córdoba
    given-names: Gonzalo
    name-suffix: Jr.
    role: tester
  ...
```

`type`, `languages`, `programming-languages`, `status`

These keys only take values from a defined set, cf. the respective sections:

- Reference types
- Language strings
- Programming language strings
- Status strings

`license-url`, `repository`, `repository-code`, `repository-artifact`, `url`

These keys take URL strings as values.

keywords

This key takes a collection of strings.

Example:

```
references:
- type: software
  keywords:
  - linguistics
  - "multi-layer annotation"
  - web service
...
```

scope

A reference item can specify a more detailed scope for the reference, via the reference key `scope`. This key can be useful if certain references should only be cited under specific circumstances, e.g., only when a specific package of the software is used. In such a case, the package would ideally have its own CFF file, but if this is not possible for whatever reason, the `scope` key may come in handy.

Example:

```
references:
- scope: "Cite this paper when you run software X with flag --xy"
  type: article
...
```

Formatting

CFF follows the formatting rules of YAML 1.2, of which one of the most important ones is that the colon (:) after a key should always be followed by a whitespace.

Structure is determined by indentation, i.e., lines containing nested elements must be indented by at least one whitespace character, although using at least two whitespaces as a standard for indentation preserves readability.

Value strings can (and sometimes should) be double-quoted, e.g. `"string"`, especially when they contain YAML special characters, or special characters in general. These include:

: { } [] , & * # ? | - < > = ! % @ \

Reference keys

CFF defines the following reference keys.

Table 1: Complete list of CFF keys.

| CFF Key | CFF Data Type | Description |
|-------------------------------|--|---|
| <code>abbreviation</code> | String | The abbreviation of the work |
| <code>abstract</code> | String | The abstract of a work |
| <code>authors</code> | Collection of <i>entity</i> or <i>person objects</i> | The author of a work |
| <code>collection-doi</code> | String | The DOI of a collection containing the work |
| <code>collection-title</code> | String | The title of a collection or proceedings |
| <code>collection-type</code> | String | The type of a collection |
| <code>commit</code> | String | The (e.g., Git) commit hash or (e.g., Subversion) revision number of the work |
| <code>conference</code> | <i>Entity object</i> | The conference where the work was presented |

| CFF Key | CFF Data Type | Description |
|-------------------|--|--|
| contact | Collection of <i>entity</i> or <i>person objects</i> | The contact person, group, company, etc. for a work |
| copyright | String | The copyright information pertaining to the work |
| data-type | String | The data type of a data set |
| database | String | The name of the database where a work was accessed/is stored |
| database-provider | <i>Entity object</i> | The provider of the database where a work was accessed/is stored |
| date-accessed | Date | The date the work has been last accessed |
| date-downloaded | Date | The date the work has been downloaded |
| date-published | Date | The date the work has been published |
| date-released | Date | The date the work has been released |
| department | String | The department where a work has been produced |
| doi | String | The DOI of the work |
| edition | String | The edition of the work |
| editors | Collection of <i>entity</i> or <i>person objects</i> | The editors of a work |
| editors-series | Collection of <i>entity</i> or <i>person objects</i> | The editors of a series in which a work has been published |
| end | Integer | The end page of the work |
| entry | String | An entry in the collection that constitutes the work |
| filename | String | The name of the electronic file containing the work |
| format | String | The format in which a work is represented |
| institution | <i>Entity object</i> | The institution where a work has been produced or published |
| isbn | String | The ISBN of the work |
| issn | String | The ISSN of the work |
| issue | Integer | The issue of a periodical in which a work appeared |
| issue-date | String | The publication date of the issue of a periodical in which a work appeared |
| issue-title | String | The name of the issue of a periodical in which the work appeared |
| journal | String | The name of the journal/magazine/newspaper/periodical where the work was published |
| keywords | Collection of strings | Keywords pertaining to the work |
| languages | Collection of ISO 639 <i>language strings</i> | The language of the work |
| license | String | The license under which a work is licensed |
| license-url | String (<i>URL</i>) | The URL of the license text under which a work is licensed |
| location | <i>Entity object</i> | The location of the work |
| loc-start | Integer | The line of code in the file where the work starts |
| loc-end | Integer | The line of code in the file where the work ends |

| CFF Key | CFF Data Type | Description |
|-----------------------|---|---|
| medium | String | The medium of the work |
| month | Integer | The month in which a work has been published |
| nihmsid | String | The NIHMSID of a work |
| notes | String | Notes pertaining to the work |
| number | String | The accession number for a work |
| number-volumes | Integer | The number of volumes making up the collection in which the work has been published |
| pages | Integer | The number of pages of the work |
| patent-states | String | The states for which a patent is granted |
| pmcid | String | The PMCID of a work |
| programming-languages | Collection of <i>programming language strings</i> | The programming language of the work |
| publisher | <i>Entity object</i> | The publisher who has published the work |
| recipients | Collection of <i>entity or person objects</i> | The recipient of a personal communication |
| repository | String (<i>URL</i>) | The repository where the work is stored |
| repository-code | String (<i>URL</i>) | The version control system where the source code of the work is stored |
| repository-artifact | String (<i>URL</i>) | The repository where the (executable/binary) artifact of the work is stored |
| scope | String | The scope of the reference, e.g., the section of the work it adheres to |
| section | String | The section of a work that is referenced |
| senders | Collection of <i>person objects</i> | The sender of a personal communication |
| status | <i>Status string</i> | The publication status of the work |
| start | Integer | The start page of the work |
| thesis-type | String | The type of the thesis that is the work |
| title | String | The title of the work |
| translators | Collection of <i>entity or person objects</i> | The translator of a work |
| type | <i>Reference types string</i> | The type of the work |
| url | String (<i>URL</i>) | The URL of the work |
| version | String | The version of the work |
| volume | Integer | The volume of the periodical in which a work appeared |
| volume-title | String | The title of the volume in which the work appeared |
| year | Integer | The year in which a work has been published |
| year-original | Integer | The year of the original publication |

Exemplary uses

This section details exemplary use cases for some of the keys to avoid ambiguity/misuse.

abstract

- If the work is a journal paper or other academic work: The abstract of the work.
- If the work is a film, broadcast or similar: The synopsis of the work.

department

- If the work is a thesis: The academic department where the thesis has been produced.
- If the work is a government document: The governmental department which has issued the document.

format

- If the work is a music file: The digital format in which a musical piece is saved, e.g., MP3.
- If the work is a data set: The digital format in which the data set is saved.
- If the work is a painting: The format of the painting, e.g., the width and height of the canvas.

institution

- If the work is a report: The institution where the report has been produced.
- If the work is a case: The court where a case has been held.
- If the work is a blog post: The institution responsible for running the blog.
- If the work is a patent, legal rule or similar: The issuing institution of the patent/rule.
- If the work is a grant: The funding agency sponsoring the grant.
- If the work is a thesis: The university where a thesis has been produced.
- If the work is a statute: The institution or geographical unit which the statute adheres to.
- If the work is a conference: The organisation which held the conference.

languages

- If the work is a book: The language in which the book is written.

location

- If the work is an artwork: E.g., the museum holding the work.
- If the work is a historical work, illuminated manuscript or similar: The library or archive where the work is held.

medium

- If the work is an artwork: The medium of the artwork, e.g., “photograph”, “painting”, “oil on canvas”, etc.
- If the work is a book or similar: Whether it is a printed book or an ebook.

month

- If the work is a conference: The month in which the conference has been held.
- If the work is a magazine article: The month in which the magazine issue containing the article has been published.

number

- If the work is a conference paper: E.g., the submission number of the paper
- If the work is a grant: The grant number provided by the funding agency.
- If the work is a work of art: E.g., the catalogue number provided by a museum holding the artwork.
- If the work is a report: The report number of a report.
- If the work is a patent: The patent number of the work.
- If the work is a historical work, illuminated manuscript or similar: The codex or folio number of a manuscript, or the library identifier for a manuscript.

term

- If the work is a dictionary or encyclopedia: The term in the dictionary or encyclopedia that is being referenced.

title

- If the work is a case: The name of the case (e.g., Name v. Name).

version

- If the work is a software: The version of the referenced software.

Reference types

Table 2: Complete list of CFF reference types.

| Reference type string | Description |
|---------------------------------|---|
| art | A work of art, e.g., a painting |
| article | |
| audiovisual | |
| bill | A legal bill |
| blog | A blog post |
| book | A book or e-book |
| catalogue | |
| conference | |
| conference-paper | |
| data | A data set |
| database | An aggregated or online database |
| dictionary | |
| edited-work | An edited work, e.g., a book |
| encyclopedia | |
| film-broadcast | A film or broadcast |
| generic | The fallback type |
| government-document | |
| grant | A research or other grant |
| hearing | |
| historical-work | A historical work, e.g., a medieval manuscript |
| legal-case | |
| legal-rule | |
| magazine-article | |
| manual | A manual |
| map | A geographical map |
| multimedia | A multimedia file |
| music | A music file or sheet music |
| newspaper-article | |
| pamphlet | |
| patent | |
| personal-communication | |
| proceedings | Conference proceedings |
| report | |
| serial | |
| slides | Slides, i.e., a published slide deck |
| software | Software |
| software-code | Software source code |
| software-container | A software container (e.g., a docker container) |
| software-executable | An executable software, i.e., a binary/artifact |
| software-virtual-machine | A virtual machine/vm image |
| sound-recording | |
| standard | |
| statute | |
| thesis | An academic thesis |
| unpublished | |
| video | A video recording |
| website | |

Objects

Entity objects

Entity objects can represent different types of entities, e.g., a publishing company, or conference. In CFF, they are realized as collections with a defined set of keys. Only the key `name` is mandatory.

Table 3: Complete list of keys for entity objects.

| Entity key | Entity Data Type | optional |
|-------------------------|-----------------------|----------|
| <code>name</code> | String | |
| <code>address</code> | String | • |
| <code>city</code> | String | • |
| <code>region</code> | String | • |
| <code>post-code</code> | String | • |
| <code>country</code> | String | • |
| <code>orcid</code> | String | • |
| <code>email</code> | String | • |
| <code>tel</code> | String | • |
| <code>fax</code> | String | • |
| <code>website</code> | String (<i>URL</i>) | • |
| <code>date-start</code> | Date | • |
| <code>date-end</code> | Date | • |
| <code>location</code> | String | • |

Exemplary uses

`address`

- To be used for street names and house numbers, etc.

`region`

- To be used for, e.g., states (as in US states or German federal states).

`post-code`

- The post code or zip code of an address.

`country`

- The ISO 3166-1 alpha-2 country code for a country. A list of ISO 3166-1 alpha-2 codes can be found at [Wikipedia:ISO 3166-1](https://en.wikipedia.org/wiki/ISO_3166-1).

Example:

`references:`

```
- type: book
  publisher:
    - name: PeerJ
      city: London
      country: GB
```

`date-start` and `date-end`

- The start and end date of, e.g., a conference. This must be formatted according to ISO 8601, e.g., `YYYY-MM-DD`, or `2017-10-04T16:20:57+00:00`.

Person objects

A person object represents a person. In CFF, person objects are realized as collections with a defined set of keys, of which only `family-names` and `given-names` are mandatory.

Table 4: Complete list of keys for person objects.

| Entity key | Entity Data Type | optional |
|----------------------------|----------------------------|----------|
| <code>family-names</code> | String | |
| <code>given-names</code> | String | |
| <code>name-particle</code> | String | • |
| <code>name-suffix</code> | String | • |
| <code>affiliation</code> | String | • |
| <code>address</code> | String | • |
| <code>city</code> | String | • |
| <code>region</code> | String | • |
| <code>post-code</code> | String | • |
| <code>country</code> | String | • |
| <code>orcid</code> | String | • |
| <code>email</code> | String | • |
| <code>tel</code> | String | • |
| <code>fax</code> | String | • |
| <code>website</code> | String (<i>URL</i>) | • |
| <code>role</code> | <i>Person roles</i> string | • |

Exemplary uses

Name keys

CFF aims at implementing a culturally neutral model for personal names, according to the suggestions on splitting personal names by the W3C and the implementation of personal name splitting in BibTeX [5].

To this end, CFF provides four generic keys to specify personal names:

1. Values for `family-names` specify family names, including combinations of given and patronymic forms, such as *Guðmundsdóttir* or *bin Osman*; double names with or without hyphen, such as *Leutheusser-Schnarrenberger* or *Sánchez Vicario*. It can potentially also specify names that include prepositions or (nobilary) particles, especially if they occur in between family names such as in Spanish- or Portuguese-origin names, such as *Fernández de Córdoba*.
2. Values for `given-names` specify given and any other names.
3. Values for `name-particle` specify nobiliary particles and prepositions, such as in Ludwig *van* Beethoven or Rafael *van der* Vaart.
4. Values for `name-suffix` specify suffixes such as *Jr.* or *III* (as in Frank Edwin Wright *III*).

Note that these keys may still not be optimal for, e.g., Icelandic names which do not have the concept of family names, or Chinese generation names, but the alternative is highly localized customization, which would be counterintuitive as to CFF's goal to be easily accessible. Thus, it is ultimately the task of CFF file authors to find the optimal name split in any given case.

affiliation

- To specify the affiliation of a person, e.g., a university, research centre, etc.

Address keys

- Cf. Entity objects for details.

orcid

- To specify an ORCID identifier in the format `dddd-dddd-dddd-dddd`, e.g., `0000-0003-4925-7248`.

Person roles

A person object can be assigned a role for the purposes of specifying authorship status, e.g., to distinguish main authors of a software from contributors who have provided a small patch. The defined roles are:

Table 5: Defined roles for person objects.

| Key |
|--|
| <code>administrator</code> (e.g., of a software system) |
| <code>artist</code> |
| <code>assignee</code> (e.g., of a patent) |
| <code>author</code> |
| <code>benchmarker</code> (e.g., of a software) |
| <code>cartographer</code> |
| <code>composer</code> |
| <code>contributor</code> |
| <code>creator</code> |
| <code>designer</code> |
| <code>director</code> (e.g., of a movie) |
| <code>editor</code> (e.g., of an edited book/edition) |
| <code>evangelist</code> (e.g., for a software) |
| <code>institution</code> (e.g., issuing a standard) |
| <code>inventor</code> |
| <code>main-author</code> |
| <code>maintainer</code> (of a software project) |
| <code>manager</code> (e.g., of a software project) |
| <code>programmer</code> |
| <code>reporter</code> (e.g., of a court case/a software bug) |
| <code>researcher</code> (e.g., authoring a data set/informing a software implementation) |
| <code>engineer</code> (e.g., for a software) |
| <code>technical-writer</code> (e.g., of a software documentation) |
| <code>tester</code> (e.g., of a software) |
| <code>trainer</code> |

Specified value strings

The keys `status`, `languages` and `programming-languages` can only take values from a fixed set of strings. These are specified below.

Status strings

Works can have a different status of publication, e.g., journal papers. CFF specifies the following value strings for the key `status`.

Table 6: Defined statuses for works.

| Status (String) | Description |
|-----------------------------|--|
| <code>in-preparation</code> | A work in preparation, e.g., a manuscript |
| <code>abstract</code> | The abstract of a work |
| <code>submitted</code> | A work that has been submitted for publication |
| <code>in-press</code> | A work that has been accepted for publication but has not yet been published |
| <code>advance-online</code> | A work that has been published online in advance of publication in the target medium |

Language strings

Natural languages as a value for the key `languages` are specified via their respective 3-character ISO 639-3 code. A list of ISO 639-3 codes is maintained at [Wikipedia:List of ISO 639-3 codes](#). Alternatively, a language's 2-character ISO 639-1 code may be used. A list of ISO 639-1 codes is maintained at [Wikipedia:List of ISO 639-1 codes](#).

Example for a work in both English and Daakaka:

```
references:
- type: book
  ...
languages:
- en
- bpa
```

Programming language strings

CFF specifies the following value strings for the key `programming-languages`. If a language is not included, please use the string `other` with a lower-case, hyphenated string argument, and do not include the version of the programming language used, e.g., for *My Fancy Language v4.2.1*, use `other=my-fancy-language`. Additionally, please create an issue on the GitHub repository for CFF, asking to include the programming language in the list.

Table 7: List of programming language names available in CFF. Table based on the languages available on GitHub (via <https://github.com/github/linguist/blob/master/lib/linguist/languages.yml>, MIT license, Copyright (c) 2017 GitHub, Inc.).

| CFF key | Language name | Language type |
|---------------------------------------|--------------------------|---------------|
| <code>1c-enterprise</code> | 1C Enterprise | programming |
| <code>abap</code> | ABAP | programming |
| <code>abnf</code> | ABNF | data |
| <code>actionscript</code> | ActionScript | programming |
| <code>ada</code> | Ada | programming |
| <code>adobe-font-metrics</code> | Adobe Font Metrics | data |
| <code>agda</code> | Agda | programming |
| <code>ags-script</code> | AGS Script | programming |
| <code>alloy</code> | Alloy | programming |
| <code>alpine-abuild</code> | Alpine Abuild | programming |
| <code>ampl</code> | AMPL | programming |
| <code>ant-build-system</code> | Ant Build System | data |
| <code>antlr</code> | ANTLR | programming |
| <code>apacheconf</code> | ApacheConf | data |
| <code>apex</code> | Apex | programming |
| <code>api-blueprint</code> | API Blueprint | markup |
| <code>apl</code> | APL | programming |
| <code>apollo-guidance-computer</code> | Apollo Guidance Computer | programming |
| <code>applescript</code> | AppleScript | programming |
| <code>arc</code> | Arc | programming |
| <code>arduino</code> | Arduino | programming |
| <code>asciidoc</code> | AsciiDoc | prose |
| <code>asn.1</code> | ASN.1 | data |
| <code>asp</code> | ASP | programming |
| <code>aspectj</code> | AspectJ | programming |
| <code>assembly</code> | Assembly | programming |
| <code>ats</code> | ATS | programming |
| <code>augeas</code> | Augeas | programming |
| <code>autohotkey</code> | AutoHotkey | programming |

| CFF key | Language name | Language type |
|-------------------|-------------------|---------------|
| autoit | AutoIt | programming |
| awk | Awk | programming |
| ballerina | Ballerina | programming |
| batchfile | Batchfile | programming |
| befunge | Befunge | programming |
| bison | Bison | programming |
| bitbake | BitBake | programming |
| blade | Blade | markup |
| blitzbasic | BlitzBasic | programming |
| blitzmax | BlitzMax | programming |
| bluespec | Bluespec | programming |
| boo | Boo | programming |
| brainfuck | Brainfuck | programming |
| brightscript | Brightscript | programming |
| bro | Bro | programming |
| c# | C# | programming |
| c++ | C++ | programming |
| c | C | programming |
| c-objdump | C-ObjDump | data |
| c2hs-haskell | C2hs Haskell | programming |
| cap'n-proto | Cap'n Proto | programming |
| cartocss | CartoCSS | programming |
| ceylon | Ceylon | programming |
| chapel | Chapel | programming |
| charity | Charity | programming |
| chuck | ChucK | programming |
| cirru | Cirru | programming |
| clarion | Clarion | programming |
| clean | Clean | programming |
| click | Click | programming |
| clips | CLIPS | programming |
| clojure | Clojure | programming |
| closure-templates | Closure Templates | markup |
| cmake | CMake | programming |
| cobol | COBOL | programming |
| coffeescript | CoffeeScript | programming |
| coldfusion | ColdFusion | programming |
| coldfusion-cfc | ColdFusion CFC | programming |
| collada | COLLADA | data |
| common-lisp | Common Lisp | programming |
| component-pascal | Component Pascal | programming |
| cool | Cool | programming |
| coq | Coq | programming |
| cpp-objdump | Cpp-ObjDump | data |
| creole | Creole | prose |
| crystal | Crystal | programming |
| cson | CSON | data |
| csound | Csound | programming |
| csound-document | Csound Document | programming |
| csound-score | Csound Score | programming |
| css | CSS | markup |
| csv | CSV | data |
| cuda | Cuda | programming |
| cweb | CWeb | programming |
| cycrypt | Cycrypt | programming |

| CFF key | Language name | Language type |
|--------------------------|--------------------------|---------------|
| cython | Cython | programming |
| d | D | programming |
| d-objdump | D-ObjDump | data |
| darcs-patch | Darcs Patch | data |
| dart | Dart | programming |
| dataweave | DataWeave | programming |
| desktop | desktop | data |
| diff | Diff | data |
| digital-command-language | DIGITAL Command Language | programming |
| dm | DM | programming |
| dns-zone | DNS Zone | data |
| dockerfile | Dockerfile | data |
| dogescript | Dogescript | programming |
| dtrace | DTrace | programming |
| dylan | Dylan | programming |
| e | E | programming |
| eagle | Eagle | data |
| easybuild | Easybuild | data |
| ebnf | EBNF | data |
| ec | eC | programming |
| ecere-projects | Ecere Projects | data |
| ecl | ECL | programming |
| eclipse | ECLIPSe | programming |
| edn | edn | data |
| eiffel | Eiffel | programming |
| ejs | EJS | markup |
| elixir | Elixir | programming |
| elm | Elm | programming |
| emacs-lisp | Emacs Lisp | programming |
| emberscript | EmberScript | programming |
| eq | EQ | programming |
| erlang | Erlang | programming |
| f# | F# | programming |
| factor | Factor | programming |
| fancy | Fancy | programming |
| fantom | Fantom | programming |
| filebench-wml | Filebench WML | programming |
| filterscript | Filterscript | programming |
| fish | fish | programming |
| flux | FLUX | programming |
| formatted | Formatted | data |
| forth | Forth | programming |
| fortran | Fortran | programming |
| freemarker | FreeMarker | programming |
| frege | Frege | programming |
| g-code | G-code | data |
| game-maker-language | Game Maker Language | programming |
| gams | GAMS | programming |
| gap | GAP | programming |
| gcc-machine-description | GCC Machine Description | programming |
| gdb | GDB | programming |
| gdscrip | GDScrip | programming |
| genie | Genie | programming |
| genshi | Genshi | programming |
| gentoo-ebuild | Gentoo Ebuild | programming |

| CFF key | Language name | Language type |
|-------------------------|-------------------------|---------------|
| gentoo-eclass | Gentoo Eclass | programming |
| gerber-image | Gerber Image | data |
| gettext-catalog | Gettext Catalog | prose |
| gherkin | Gherkin | programming |
| gls1 | GLSL | programming |
| glyph | Glyph | programming |
| gn | GN | data |
| gnuplot | Gnuplot | programming |
| go | Go | programming |
| golo | Golo | programming |
| gosu | Gosu | programming |
| grace | Grace | programming |
| gradle | Gradle | data |
| grammatical-framework | Grammatical Framework | programming |
| graph-modeling-language | Graph Modeling Language | data |
| graphql | GraphQL | data |
| graphviz | Graphviz (DOT) | data |
| groovy | Groovy | programming |
| groovy-server-pages | Groovy Server Pages | programming |
| hack | Hack | programming |
| haml | Haml | markup |
| handlebars | Handlebars | markup |
| harbour | Harbour | programming |
| haskell | Haskell | programming |
| haxe | Haxe | programming |
| hcl | HCL | programming |
| hls1 | HLSL | programming |
| html+django | HTML+Django | markup |
| html+ecr | HTML+ECR | markup |
| html+eex | HTML+EEX | markup |
| html+erb | HTML+ERB | markup |
| html+php | HTML+PHP | markup |
| html | HTML | markup |
| http | HTTP | data |
| hy | Hy | programming |
| hyphy | HyPhy | programming |
| idl | IDL | programming |
| idris | Idris | programming |
| igor-pro | IGOR Pro | programming |
| inform-7 | Inform 7 | programming |
| ini | INI | data |
| inno-setup | Inno Setup | programming |
| io | Io | programming |
| ioke | Ioke | programming |
| irc-log | IRC log | data |
| isabelle | Isabelle | programming |
| isabelle-root | Isabelle ROOT | programming |
| j | J | programming |
| jasmin | Jasmin | programming |
| java | Java | programming |
| java-server-pages | Java Server Pages | programming |
| javascript | JavaScript | programming |
| jflex | JFlex | programming |
| jison | Jison | programming |
| jison-lex | Jison Lex | programming |

| CFF key | Language name | Language type |
|-----------------------|-----------------------|---------------|
| jolie | Jolie | programming |
| json | JSON | data |
| json5 | JSON5 | data |
| jsoniq | JSONiq | programming |
| jsonld | JSONLD | data |
| jsx | JSX | programming |
| julia | Julia | programming |
| jupyter-notebook | Jupyter Notebook | markup |
| kicad-layout | KiCad Layout | data |
| kicad-legacy-layout | KiCad Legacy Layout | data |
| kicad-schematic | KiCad Schematic | data |
| kit | Kit | markup |
| kotlin | Kotlin | programming |
| krl | KRL | programming |
| labview | LabVIEW | programming |
| lasso | Lasso | programming |
| latte | Latte | markup |
| lean | Lean | programming |
| less | Less | markup |
| lex | Lex | programming |
| lfe | LFE | programming |
| lilypond | LilyPond | programming |
| limbo | Limbo | programming |
| linker-script | Linker Script | data |
| linux-kernel-module | Linux Kernel Module | data |
| liquid | Liquid | markup |
| literate-agda | Literate Agda | programming |
| literate-coffeescript | Literate CoffeeScript | programming |
| literate-haskell | Literate Haskell | programming |
| livescript | LiveScript | programming |
| llvm | LLVM | programming |
| logos | Logos | programming |
| logtalk | Logtalk | programming |
| lolcode | LOLCODE | programming |
| lookml | LookML | programming |
| loomscript | LoomScript | programming |
| lsl | LSL | programming |
| lua | Lua | programming |
| m | M | programming |
| m4 | M4 | programming |
| m4sugar | M4Sugar | programming |
| makefile | Makefile | programming |
| mako | Mako | programming |
| markdown | Markdown | prose |
| marko | Marko | markup |
| mask | Mask | markup |
| mathematica | Mathematica | programming |
| matlab | Matlab | programming |
| maven-pom | Maven POM | data |
| max | Max | programming |
| maxscript | MAXScript | programming |
| mediawiki | MediaWiki | prose |
| mercury | Mercury | programming |
| meson | Meson | programming |
| metal | Metal | programming |

| CFF key | Language name | Language type |
|--------------------------|--------------------------|---------------|
| minid | MiniD | programming |
| mirah | Mirah | programming |
| modelica | Modelica | programming |
| modula-2 | Modula-2 | programming |
| module-management-system | Module Management System | programming |
| monkey | Monkey | programming |
| moocode | Moocode | programming |
| moonscript | MoonScript | programming |
| mql4 | MQL4 | programming |
| mql5 | MQL5 | programming |
| mtml | MTML | markup |
| muf | MUF | programming |
| mupad | mupad | programming |
| myghty | Myghty | programming |
| ncl | NCL | programming |
| nearley | Nearley | programming |
| nemerle | Nemerle | programming |
| nesc | nesC | programming |
| netlinx+erb | NetLinx+ERB | programming |
| netlinx | NetLinx | programming |
| netlogo | NetLogo | programming |
| newlisp | NewLisp | programming |
| nginx | Nginx | data |
| nim | Nim | programming |
| ninja | Ninja | data |
| nit | Nit | programming |
| nix | Nix | programming |
| nl | NL | data |
| nsis | NSIS | programming |
| nu | Nu | programming |
| numpy | NumPy | programming |
| objdump | ObjDump | data |
| objective-c++ | Objective-C++ | programming |
| objective-c | Objective-C | programming |
| objective-j | Objective-J | programming |
| ocaml | OCaml | programming |
| omgrofl | Omgrofl | programming |
| ooc | ooc | programming |
| opa | Opa | programming |
| opal | Opal | programming |
| opencl | OpenCL | programming |
| openedge-abl | OpenEdge ABL | programming |
| openrc-runscrip | OpenRC runscrip | programming |
| openscad | OpenSCAD | programming |
| opentype-feature-file | OpenType Feature File | data |
| org | Org | prose |
| other | | |
| ox | Ox | programming |
| oxygene | Oxygene | programming |
| oz | Oz | programming |
| p4 | P4 | programming |
| pan | Pan | programming |
| papyrus | Papyrus | programming |
| parrot | Parrot | programming |
| parrot-assembly | Parrot Assembly | programming |

| CFF key | Language name | Language type |
|--------------------------------|--------------------------------|---------------|
| parrot-internal-representation | Parrot Internal Representation | programming |
| pascal | Pascal | programming |
| pawn | PAWN | programming |
| pep8 | Pep8 | programming |
| perl | Perl | programming |
| perl-6 | Perl 6 | programming |
| php | PHP | programming |
| pic | Pic | markup |
| pickle | Pickle | data |
| picolisp | PicoLisp | programming |
| piglatin | PigLatin | programming |
| pike | Pike | programming |
| plpgsql | PLpgSQL | programming |
| plsql | PLSQL | programming |
| pod | Pod | prose |
| pogoscript | PogoScript | programming |
| pony | Pony | programming |
| postscript | PostScript | markup |
| pov-ray-sdl | POV-Ray SDL | programming |
| powerbuilder | PowerBuilder | programming |
| powershell | PowerShell | programming |
| processing | Processing | programming |
| prolog | Prolog | programming |
| propeller-spin | Propeller Spin | programming |
| protocol-buffer | Protocol Buffer | data |
| public-key | Public Key | data |
| pug | Pug | markup |
| puppet | Puppet | programming |
| pure-data | Pure Data | data |
| purebasic | PureBasic | programming |
| purescript | PureScript | programming |
| python | Python | programming |
| python-console | Python console | programming |
| python-raise | Python raise | data |
| python-traceback | Python traceback | data |
| qmake | QMake | programming |
| qml | QML | programming |
| r | R | programming |
| racket | Racket | programming |
| ragel | Ragel | programming |
| raml | RAML | markup |
| rascal | Rascal | programming |
| raw-token-data | Raw token data | data |
| rdoc | RDoc | prose |
| realbasic | REALbasic | programming |
| reason | Reason | programming |
| rebol | Rebol | programming |
| red | Red | programming |
| redcode | Redcode | programming |
| regular-expression | Regular Expression | data |
| ren'py | Ren'Py | programming |
| renderscript | RenderScript | programming |
| restructuredtext | reStructuredText | prose |
| rexx | REXX | programming |
| rhtml | RHTML | markup |
| ring | Ring | programming |

| CFE key | Language name | Language type |
|----------------------|----------------------|---------------|
| rmarkdown | RMarkdown | prose |
| robotframework | RobotFramework | programming |
| roff | Roff | markup |
| rouge | Rouge | programming |
| rpm-spec | RPM Spec | data |
| ruby | Ruby | programming |
| runoff | RUNOFF | markup |
| rust | Rust | programming |
| sage | Sage | programming |
| saltstack | SaltStack | programming |
| sas | SAS | programming |
| sass | Sass | markup |
| scala | Scala | programming |
| scaml | Scaml | markup |
| scheme | Scheme | programming |
| scilab | Scilab | programming |
| scss | SCSS | markup |
| self | Self | programming |
| shaderlab | ShaderLab | programming |
| shell | Shell | programming |
| shellsession | ShellSession | programming |
| shen | Shen | programming |
| slash | Slash | programming |
| slim | Slim | markup |
| smali | Smali | programming |
| smalltalk | Smalltalk | programming |
| smarty | Smarty | programming |
| smt | SMT | programming |
| sourcepawn | SourcePawn | programming |
| sparql | SPARQL | data |
| spline-font-database | Spline Font Database | data |
| sqf | SQF | programming |
| sql | SQL | data |
| sqlpl | SQLPL | programming |
| squirrel | Squirrel | programming |
| srcode-template | SRecode Template | markup |
| stan | Stan | programming |
| standard-ml | Standard ML | programming |
| stata | Stata | programming |
| ston | STON | data |
| stylus | Stylus | markup |
| sublime-text-config | Sublime Text Config | data |
| subrip-text | SubRip Text | data |
| supercollider | SuperCollider | programming |
| svg | SVG | data |
| swift | Swift | programming |
| systemverilog | SystemVerilog | programming |
| tcl | Tcl | programming |
| tcsh | Tcsh | programming |
| tea | Tea | markup |
| terra | Terra | programming |
| tex | TeX | markup |
| text | Text | prose |
| textile | Textile | prose |
| thrift | Thrift | programming |

| CFF key | Language name | Language type |
|------------------------------|------------------------------|---------------|
| ti-program | TI Program | programming |
| tla | TLA | programming |
| toml | TOML | data |
| turing | Turing | programming |
| turtle | Turtle | data |
| twig | Twig | markup |
| txl | TXL | programming |
| type-language | Type Language | data |
| typescript | TypeScript | programming |
| unified-parallel-c | Unified Parallel C | programming |
| unity3d-asset | Unity3D Asset | data |
| unix-assembly | Unix Assembly | programming |
| uno | Uno | programming |
| unrealscript | UnrealScript | programming |
| urweb | UrWeb | programming |
| vala | Vala | programming |
| vcl | VCL | programming |
| verilog | Verilog | programming |
| vhdl | VHDL | programming |
| vim-script | Vim script | programming |
| visual-basic | Visual Basic | programming |
| volt | Volt | programming |
| vue | Vue | markup |
| wavefront-material | Wavefront Material | data |
| wavefront-object | Wavefront Object | data |
| web-ontology-language | Web Ontology Language | data |
| webassembly | WebAssembly | programming |
| webidl | WebIDL | programming |
| wisp | wisp | programming |
| world-of-warcraft-addon-data | World of Warcraft Addon Data | data |
| x10 | X10 | programming |
| xbase | xBase | programming |
| xc | XC | programming |
| xcompose | XCompose | data |
| xml | XML | data |
| xojo | Xojo | programming |
| xpages | XPages | data |
| xpm | XPM | data |
| xproc | XProc | programming |
| xquery | XQuery | programming |
| xs | XS | programming |
| xslt | XSLT | programming |
| xtend | Xtend | programming |
| yacc | Yacc | programming |
| yaml | YAML | data |
| yang | YANG | data |
| zephir | Zephir | programming |
| zimpl | Zimpl | programming |

Schema

Work is still in progress to provide a schema for CFF, against which CFF files can be validated.

Examples

Software examples

One of the main foci of CFF is to comprehensively cover the provision of citation metadata for software. To this end, it should always be used based on the Software Citation Principles [1]! Please make sure you follow the best practices wherever possible. Two typical scenarios for software citation metadata include the existence and respectively lack of a DOI for the software for which citation metadata is provided, for both of which examples follow.

A software with a DOI

Note that [1, p. 12] recommend

[...] the use of DOIs as the unique identifier due to their common usage and acceptance, particularly as they are the standard for other digital products such as publications.

Furthermore, DOIs should point to a “unique, specific software version” [1, p. 12]. Also it is recommended [1, p. 13] that:

the [DOI] should resolve to a persistent landing page that contains metadata and a link to the software itself, rather than directly to the source code files, repository, or executable.

Therefore, a minimal CITATION.cff file in such a case would look similar to the following.

```
cff-version: 1.0.0
message: If you use this software, please cite it as below.
references:
- type: software
  authors:
    - family-names: Druskat
      given-names: Stephan
      orcid: 0000-0003-4925-7248
  title: My Research Tool
  version: 1.0.4
  doi: 10043/zenodo.1234
```

A more comprehensive version could look similar to the following.

```
cff-version: 1.0.0
message: If you use this software, please cite it as below.
references:
- type: software
  authors:
    - family-names: Druskat
      given-names: Stephan
      orcid: 0000-0003-4925-7248
      affiliation: "Humboldt-Universität zu Berlin, Dept. of German Studies
        and Linguistics"
      email: mail@sdruskat.net
      website: https://hu.berlin/sdruskat
  title: My Research Tool
  version: 1.0.4
  doi: 10043/zenodo.1234
  repository-code: https://github.com/sdruskat/my-research-tool
  repository-artifact: https://hu.berlin/nexus/mrt
  date-published: 2017-09-23
  keywords:
    - "McAuthor's algorithm"
    - linguistics
```

```

- nlp
- parser
- deep convolutional neural network
programming-languages:
- java
- python
- c
- haskell
- pascal
- rust
license: Apache License, Version 2.0
license-url: http://www.apache.org/licenses/LICENSE-2.0
url: https://sdruskat.github.io/my-research-tool

```

A software without a DOI

For software without a DOI, it is recommended that “the metadata should still provide information on how to access the specific software, but this may be a company’s product number or a link to a website that allows the software be purchased.” [1, p. 13]. Furthermore, “if the version number and release date are not available, the download date can be used. Similarly, the contact name/email is an alternative to the location/repository.” [1, p. 7]

Hence, for a closed source software without a DOI for which the version number and release date cannot be determined, a CITATION.cff file could look like this.

```

cff-version: 1.0.0
message: "If you dare to use this commercial, closed-source, unversioned software
in your research, please at least cite it as below."
references:
- type: software
  title: Opaquity
  number: opq-1234-XZVF-ACME-RLY
  date-downloaded: 2017-02-31
  contact:
    - family-names: Vader
      given-names: Darth
      affiliation: Dark Side Software
      location: DS-1 Orbital Battle Station, near Scarif
      email: father@imperial-empire.com
      tel: +850 (0)123-45-666

```

software (with two references)

```

cff-version: 1.0.0
message: "If you use My Research Tool, please cite both the software and the
outline paper."
references:
- type: software
  authors:
    - family-names: Doe
      given-names: Jane
      role: main-author
    - family-names: Bielefeld
      name-particle: von
      given-names: Arthur
      role: tester
    - family-names: McAuthor

```

```

    given-names: Juniper
    name-suffix: Jr.
    role: maintainer
title: My Research Tool
doi: 10043/zenodo.1234
- type: article
authors:
  - family-names: Doe
    given-names: Jane
    role: main-author
  - family-names: Bielefeld
    name-particle: von
    given-names: Arthur
    role: author
title: "My Research Tool: A 100% accuracy syntax parser for all languages"
year: 2009
journal: Journal of Hard Science Fiction
volume: 42
issue: 13
doi: 10.9999/hardscifi-lang.42132

```

software-code (without a DOI: code repository + commit)

We recognize that there are certain situations where it may not be possible to follow the recommended best-practice. For example, if (1) the software authors did not register a DOI and/or release a specific version, or (2) the version of the software used does not match what is available to cite. In those cases, falling back on a combination of the repository URL and version number/commit hash would be an appropriate way to cite the software used. [1, p. 12]

```

cff-version: 1.0.0
message: "If you use this MRT alpha snapshot version, please cite."
references:
  - type: software-code
    authors:
      - family-names: Doe
        given-names: Jane
    title: My Research Tool Prototype
    version: 0.0.1-alpha1-build1507284872
    repository-code: https://github.com/does/mrt
    commit: 160d54f9e935c914df38c1ffda752112b5c979a8

```

software-container

```

cff-version: 1.0.0
message: "If you use the MRT Docker container, please cite the following."
references:
  - type: software-container
    authors:
      - name: "Humboldt-Universität zu Berlin"
        website: https://www.linguistik.hu-berlin.de/
        role: maintainer
      - family-names: Doe
        given-names: Jane
        role: main-author
    title: mrt-iain-m-banks
    version: 1.0.4 (Iain M. Banks)

```



```
url: https://github.com/doe/docker-brew-mrt-core/blob/160d54f9e935/iain/Dockerfile
repository: https://hub.docker.hu-berlin.de/\_/mrt-iain-m-banks/
```

software-executable

```
cff-version: 1.0.0
message: "If you use MRT, please cite the following."
references:
- type: software-executable
  authors:
    - family-names: Doe
      given-names: Jane
      role: main-author
  title: My Research Tool Kickstarter
  version: 2.0.0
  doi: 10043/zenodo.1234
  repository-artifact: https://hu.berlin/nexus/mrt-kickstarter
  filename: mrt2-kickstarter.exe
```

Other examples

art

```
cff-version: 1.0.0
message: "If you use this software, please cite the following."
references:
- type: art
  authors:
    - family-names: Picasso
      given-names: Pablo
  title: Guernica
  year: 1937
  medium: Oil on canvas
  format: 349.3cm x 776.6cm
  location:
    - name: Museo Reina Sofia
      city: Madrid
      country: ES
```

article

```
cff-version: 1.0.0
message: "If you use this software, please cite the following paper."
references:
- type: article
  authors:
    - family-names: Smith
      given-names: Arfon M.
      role: main-author
    - family-names: Katz
      given-names: Daniel S.
  affiliation: "National Center for Supercomputing Applications &
  Electrical and Computer Engineering Department & School of Information
  Sciences, University of Illinois at Urbana-Champaign, Urbana, Illinois,
  United States"
```

```
  orcid: 0000-0001-5934-7525
  role: main-author
- family-names: Niemeyer
  given-names: Kyle E.
  role: main-author
- name: "FORCE11 Software Citation Working Group"
  website: https://www.force11.org/group/software-citation-working-group
title: "Software citation principles"
year: 2016
journal: PeerJ Computer Science
volume: 2
issue: e86
doi: 10.7717/peerj-cs.86
url: https://doi.org/10.7717/peerj-cs.86
```

blog

```
cff-version: 1.0.0
message: "If you use MRT in your research, please cite the following blog article."
references:
- type: blog
  authors:
    - family-names: Doe
      given-names: Jane
  title: "Implement a 100% accuracy syntax parser for all languages? No probs!"
  date-published: 2017-09-23
  url: https://hu-berlin.de/blogs/jdoe/2017/09/23/if-only
  institution:
    - name: "Humboldt-Universität zu Berlin"
      city: Berlin
      country: DE
```

book

```
cff-version: 1.0.0
message: "If you use MRT for your research, please cite the following book."
references:
- type: book
  authors:
    - family-names: Doe
      given-names: Jane
      role: main-author
  title: "The future of syntax parsing"
  year: 2017
  publisher:
    - name: Far Out Publications
      city: Bielefeld
  medium: print
```

conference-paper

```
cff-version: 1.0.0
message: "If you use MRT for your research, please cite the following."
references:
- type: conference-paper
```

```
authors:
  - family-names: Doe
    given-names: Jane
title: "Ultimate-accuracy syntax parsing with My Research Tool"
year: 2017
collection-title: "Proceedings of the 1st Conference on Wishful Thinking"
collection-doi: 10043.zenodo.4321
editors:
  - family-names: Kirk
    given-names: James T.
conference:
  - name: 1st Conference on Wishful Thinking
    location: Spock's Inn Hotel and Bar
    address: 123 Main St
    city: Bielefeld
    region: Jarvis Island
    post-code: 12345
    country: UM
    date-start: 2017-04-01
    date-end: 2017-04-01
start: 42
end: 45
doi: 10043/zenodo.1234
```

edited-work

Note that the editors of the edited work must be specified under the `authors` key. Specific citation styles may or may not attach a suffix to the authors, such as “, eds.” or similar.

```
cff-version: 1.0.0
message: "If you use MRT, please cite the following."
references:
  - type: edited-work
    authors:
      - family-names: Doe
        given-names: Jane
    title: "Ultimate-accuracy parsing in practice"
    year: 2017
    publisher:
      - name: Far Out Publications
        city: Bielefeld
        country: DE
```

report

```
cff-version: 1.0.0
message: "If you use MRT in your research, please cite the following."
references:
  - type: report
    authors:
      - name: Fictional Parsing Interest Group, ACME Inc.
    title: "100% accuracy syntax parsing at ACME"
    url: http://www.acme.com/sigs/fp/reports/hpsp.pdf
    year: 2017
    date-accessed: 2017-09-23
```

thesis

```
cff-version: 1.0.0
message: "If you use MRT in your research, please cite the following."
references:
- type: thesis
  authors:
    - family-names: Doe
      given-names: Jane
  title: "A high accuracy syntax parser in Visual Basic"
  thesis-type: PhD
  year: 2017
  department: Dept. of Universal Language Philosophy
  institution:
    - name: "Humboldt-Universität zu Berlin"
      city: Berlin
      country: DE
  database: Thesiserver
  date-accessed: 2017-09-23
  date-published: 2017-03-21
  url: http://thesiserver.hu-berlin.de/2017/march/phd/doe-12345
```

Infrastructure

The roadmap for CFF plans for the provision of further infrastructure (e.g., software packages and web services), to support the following use cases for CFF:

- Creating CFF files
- Reading CFF files
- Validating CFF files
- Converting CFF files

Contributions

Contributions to the format specifications are welcome! For details on how to contribute, please refer to the GitHub repository for CFF at <http://github.com/sdruskat/citation-file-format>.

License

This document is licensed under a CC-BY- SA-4.0 license. The full license text can be obtained from the URL <https://creativecommons.org/licenses/by-sa/4.0/legalcode>.

References

- [1] A. M. Smith, D. S. Katz, K. E. Niemeyer, and FORCE11 Software Citation Working Group, "Software citation principles," *PeerJ Computer Science*, vol. 2, p. e86, Sep. 2016 [Online]. Available: <https://doi.org/10.7717/peerj-cs.86>
- [2] S. Druskat, "Track 2 Lightning Talk: Should CITATION files be standardized?" in *Proceedings of the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1)*, 2017 [Online]. Available: <https://doi.org/10.6084/m9.figshare.3827058>

- [3] R. Wilson, “Encouraging citation of software - introducing CITATION files.” 2013 [Online]. Available: <https://www.software.ac.uk/blog/2013-09-02-encouraging-citation-software-introducing-citation-files>
- [4] O. Ben-Kiki, C. Evans, and I. döt Net, “YAML Ain’t Markup Language (YAML™) Version 1.2. 3rd Edition, Patched at 2009-10-01.” 2009 [Online]. Available: <http://yaml.org/spec/1.2/spec.html>
- [5] J.-M. Hufflen, “Names in bibtex and mlBibTeX,” *TUGboat*, vol. 27, no. 2, pp. 243–253, Nov. 2006 [Online]. Available: <https://www.tug.org/TUGboat/tb27-2/tb87hufflen.pdf>