Archiving And Referencing All Software Source Code

Roberto Di Cosmo

February 19th, 2020

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Outline

1. Introduction
2. Academia's evolving practice
3. Archiving and referencing *all* the source code: Software Heritage
4. Zoom on the SWH-ID
5. Practical guidelines for archiving and referencing
6. What about metadata and citation?

Short Bio: Roberto Di Cosmo

Computer Science professor in Paris, now working at INRIA

- *30 years* of research (Theor. CS, Programming, Software Engineering, Erdos #: 3)
- *20 years* of Free and Open Source Software
- *10 years* building and directing structures for the common good

1999	*DemoLinux* – first live GNU/Linux distro
2007	*Free Software Thematic Group* 150 members 40 projects 200Me
2008	*Mancoosi project* www.mancoosi.org
2010	*IRILL* www.irill.org
2015	*Software Heritage* at INRIA
2018	*National Committee for Open Science*, France

The knowledge is in the *source code*

“The source code for a work means the preferred form of the work for making modifications to it."
GPL Licence

Hello World

Program (excerpt of binary)

```
4004e6:  55
4004e7:  48 89 e5
4004ea:  bf 84 05 40 00
4004ef:  b8 00 00 00 00
4004f4:  e8 c7 fe ff ff
4004f9:  90
4004fa:  5d
4004fb:  c3
```

Program (source code)

```
/* Hello World program */

#include<stdio.h>

void main()
{
    printf("Hello World");
}
```

Source code is *special*

Executable and *human readable* knowledge copyright law

"Programs must be written for people to read, and only incidentally for machines to execute."

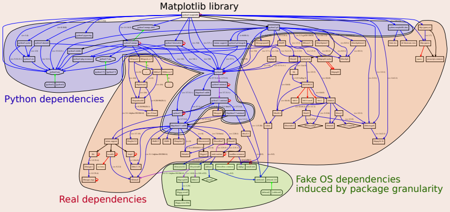
Harold Abelson

Software *evolves* over time

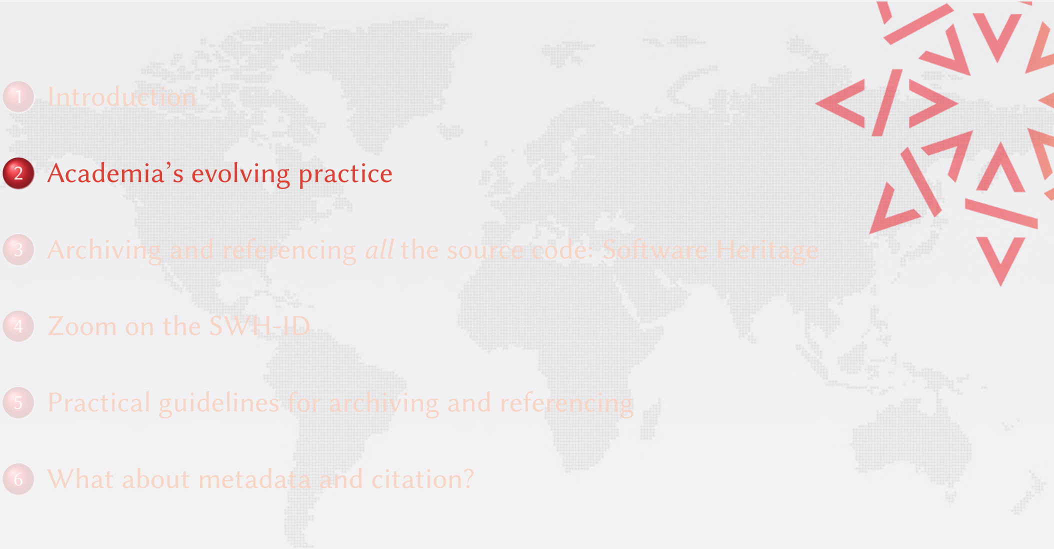
- projects may last decades
- the *development history* is key to its *understanding*

Complexity

- *millions* of lines of code
- large *web of dependencies*
  - easy to break, difficult to maintain
- sophisticated *developer communities*



Outline



1. Introduction

2. Academia's evolving practice

3. Archiving and referencing *all* the source code: Software Heritage

4. Zoom on the SWH-ID

5. Practical guidelines for archiving and referencing

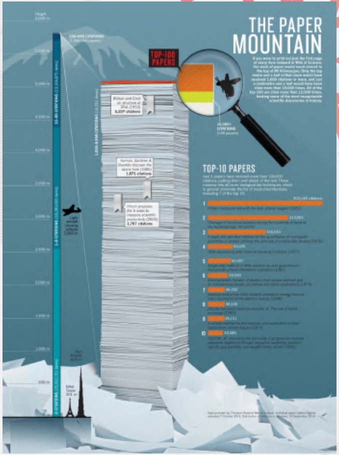
6. What about metadata and citation?

Software is a pillar of Science . . .

Software is everywhere in modern science

[. . .] the vast majority describe [. . .] or software that have become essential in their fields.

Top 100 papers (Nature, 2014)



Sometimes, if you dont have the software, you dont have the data

Christine Borgman, Paris, 2018

… a *forgotten* pillar of Open Science

Lack of recognition

not (yet) a first class citizen

- in the EOSC plan
- in the scholarly world

Lack of consensus on how to

- *archive* software
- *choose* a license
- *cite* a software project

Pressure to make the source code available is raising

Why

Necessary to…

- *reproduce* and *verify*,
- *modify* and *evolve*, building new experiments from old ones

When and where

- debate started end of first 2000 decade (bio, statistics, medicine…)
- growing in Computer Science since the ESEC/FSE 2011 Artifact Evaluation Award

A wealth of initiatives…

- Policies: ACM Artifact Review and Badging, AEC, …
- Working groups: FORCE11, RDA, SPSO, …
- Journals: IPOL, ReScience, InsightJournal, JOSS, eLife, ACM DL, …
- Repositories: FigShare, Zenodo, …
- Common infrastructures: Software Heritage

What is at stake

in increasing order of difficulty

Archival

Research software artifacts must be properly archived

make it sure we can *retrieve* them (*reproducibility*)

Identification

Research software artifacts must be properly referenced

make it sure we can *identify* them (*reproducibility*)

Metadata

Research software artifacts must be properly described

make it easy to *discover* them (*visibility*)

Citation

Research software artifacts must be properly cited *(not the same as referenced!)*

to give *credit* to authors (*evaluation*!)

Let's focus on the *first two!*

Outline

1. Introduction
2. Academia's evolving practice
3. Archiving and referencing *all* the source code: Software Heritage
4. Zoom on the SWH-ID
5. Practical guidelines for archiving and referencing
6. What about metadata and citation?

Software Heritage in a nutshell

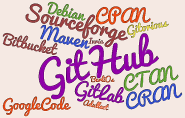
Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share the *source code* of *all the software*

Preserving our heritage, enabling better software and better science for all

Reference catalog



find and reference all the source code

Universal archive

damage
disaster
storage
reference
deletion
media
malicious
obsolete
aging
dependencies
tear
attack
dangling
wear
corruption
encryption
format

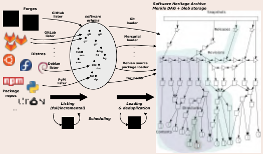
preserve all the source code

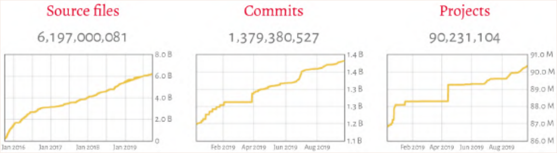
Research infrastructure

enable analysis of all the source code

Software Heritage: a revolutionary infrastructure bit.ly/swhpaper

All the software source code





The largest software source code archive *ever*

Uniform and intrinsic identifiers for reproducibility

Tracking over *20 billion software artifacts*, and counting... bit.ly/swhpidpaper

Adoption highlights

- Wikidata https://www.wikidata.org/wiki/Property:P6138
- reference archive for *swmath.org*, HAL, etalab
- part of the french National Plan for Open Science

Outline

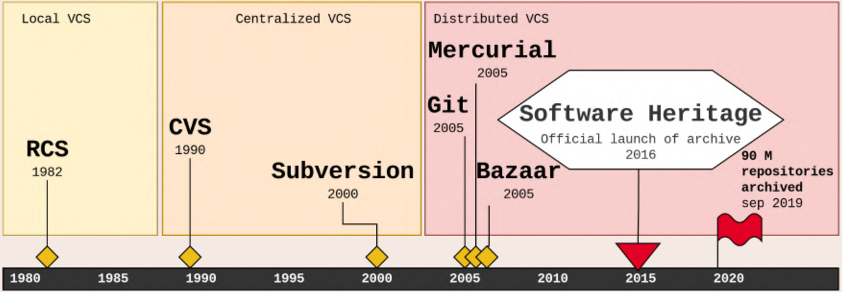
1. Introduction
2. Academia's evolving practice
3. Archiving and referencing *all* the source code: Software Heritage
4. Zoom on the SWH-ID
5. Practical guidelines for archiving and referencing
6. What about metadata and citation?

Modern software development

Version control system (VCS)

- records changes made to a (set of) *source code file* (s)
- allows to operate on versions: diff/merge/fork/recover etc.
- essential tool for software development

Three decades of evolution

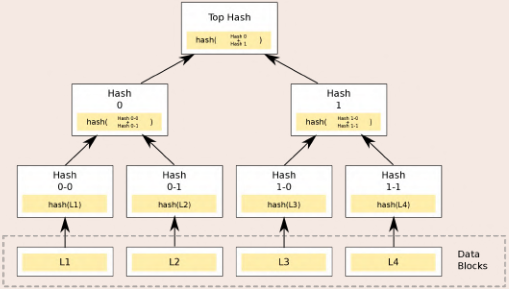


Intrinsic identifiers for modern software development

Requirements for the D in DVCS

- intrinsic unique identifiers… (here: *cryptographic signature*, aka "hash")
- … that work for tree structures (software directories)

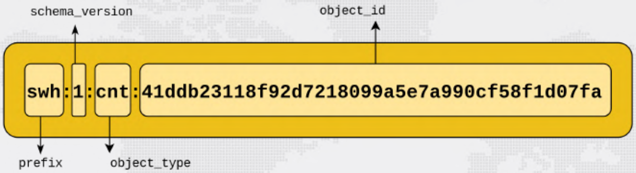
Merkle tree to the rescue (R. C. Merkle, Crypto 1979)



Combination of

- tree
- hash function

The SWH-ID schema: syntax and semantics



schema_version

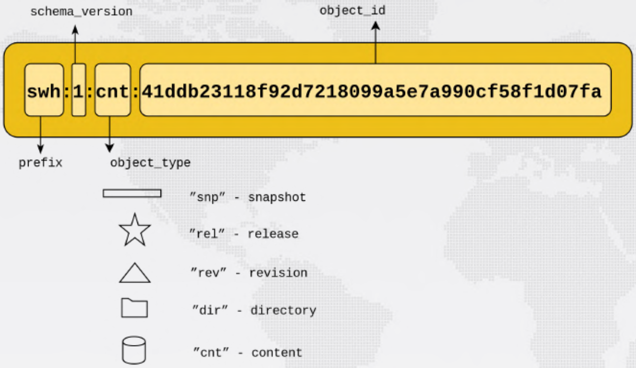
object_id

swh:1:cnt:41ddb23118f92d7218099a5e7a990cf58f1d07fa

prefix

object_type

The SWH-ID schema: syntax and semantics



schema_version

object_id

swh:1:cnt:41ddb23118f92d7218099a5e7a990cf58f1d07fa

prefix

object_type

▭ "snp" - snapshot

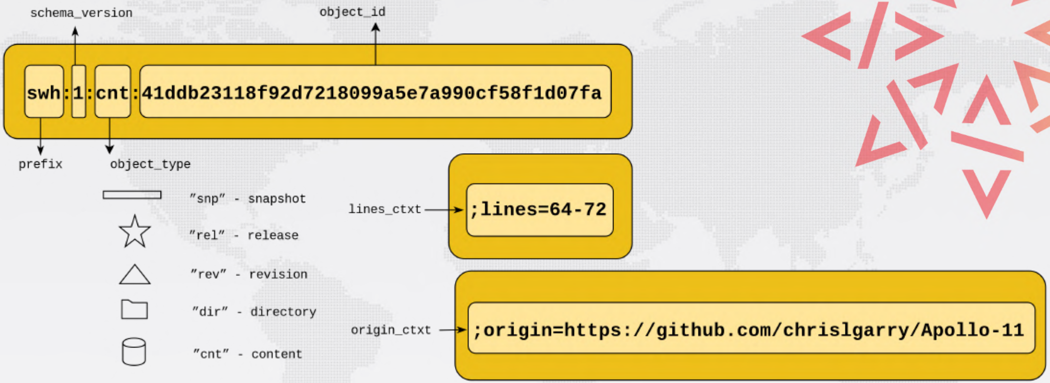
☆ "rel" - release

△ "rev" - revision

🗀 "dir" - directory

⛁ "cnt" - content

The SWH-ID schema: syntax and semantics



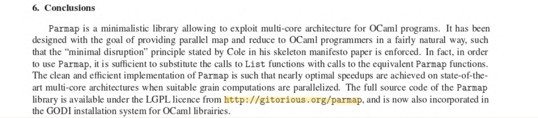
swh:1:cnt:41ddb23118f92d7218099a5e7a990cf58f1d07fa

;lines=64-72

;origin=https://github.com/chrislgarry/Apollo-11

Walkthrough the Parmap article

Danelutto and Di Cosmo, 2012

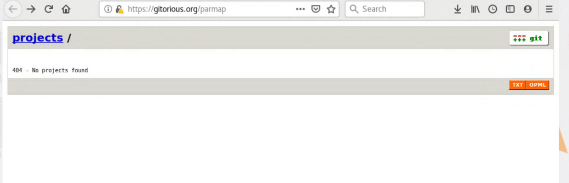


6. Conclusions

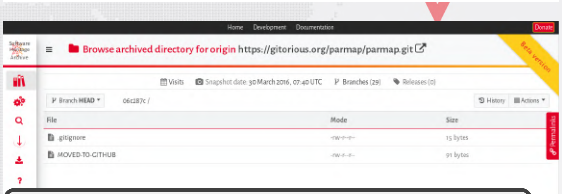
Parmap is a minimalistic library allowing to exploit multi-core architecture for OCaml programs. It has been designed with the goal of providing parallel map and reduce to OCaml programmers in a fairly natural way, such that the "minimal disruption" principle stated by Cole in his skeleton manifesto paper is enforced. In fact, in order to use Parmap, it is sufficient to substitute the calls to List functions with calls to the equivalent Parmap functions. The clean and efficient implementation of Parmap is such that nearly optimal speedups are achieved on state-of-the-art multi-core architectures when suitable grain computations are parallelized. The full source code of the Parmap library is available under the LGPL licence from http://gitorious.org/parmap, and is now also incorporated in the GODI installation system for OCaml librairies.

M. Danelutto and R. Di Cosmo, /A "Minimal Disruption" skeleton experiment: Seamless map & reduce embedding in OCaml," Procedia CS, vol. 9, pp. 1837–1846, 2012. [Online]. Available: [DOI: 10.1016/j.procs.2012.04.202]

Accessed on the 6th of February 2020



404 - No projects found



Browse archived directory for origin https://gitorious.org/parmap/parmap.git

Snapshot date: 30 March 2016, 07:40 UTC

Branches (29)

Releases (0)

Branch: HEAD

06c2B7c /

File	Mode	Size
.gitignore	-rw-r--r--	15 bytes
MOVED-TO-GITHUB	-rw-r--r--	91 bytes

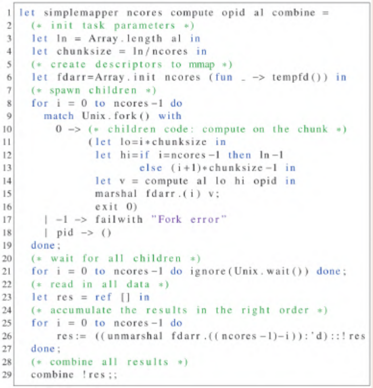
swh:1:snp:78209702559384ee1b5586df13eca84a5123aa82

Only 8 years later !

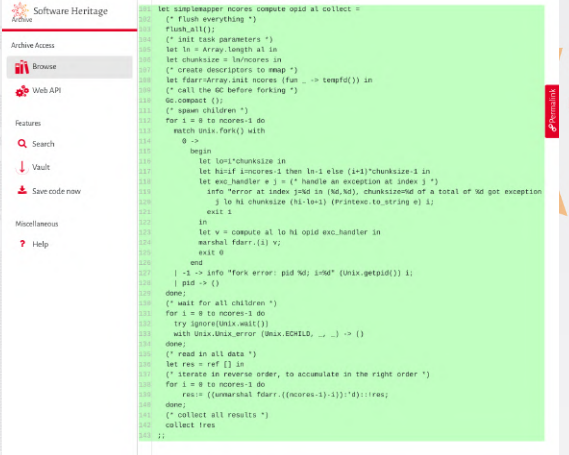
an algorithm in the source code

Figure 1 in [Danelutto and Di Cosmo, 2012]

Parmap's implementation of the distribution, fork, and recollection phase



```
let simplemapper ncores compute opid al combine =
  (* init task parameters *)
  let ln = Array.length al in
  let chunksize = ln/ncores in
  (* create descriptors to mmap *)
  let fdarr=Array.init ncores (fun _ -> tempfd()) in
  (* spawn children *)
  for i = 0 to ncores-1 do
    match Unix.fork() with
      0 -> (* children code: compute on the chunk *)
           (let lo=i*chunksize in
            let hi=if i=ncores-1 then ln-1
                   else (i+1)*chunksize-1 in
            let v = compute al lo hi opid in
            marshal fdarr.(i) v;
            exit 0)
    | -1 -> failwith "Fork error"
    | pid -> ()
  done;
  (* wait for all children *)
  for i = 0 to ncores-1 do ignore(Unix.wait()) done;
  (* read in all data *)
  let res = ref [] in
  (* accumulate the results in the right order *)
  for i = 0 to ncores-1 do
    res:= ((unmarshal fdarr.((ncores-1)-i)):'d)::!res
  done;
  (* combine all results *)
  combine !res;;
```



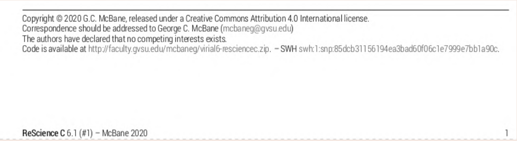
```
101 let simplemapper ncores compute opid al collect =
102   (* flush everything *)
103   flush_all();
104   (* init task parameters *)
105   let ln = Array.length al in
106   let chunksize = ln/ncores in
107   (* create descriptors to mmap *)
108   let fdarr=Array.init ncores (fun _ -> tempfd()) in
109   (* call the GC before forking *)
110   Gc.compact ();
111   (* spawn children *)
112   for i = 0 to ncores-1 do
113     match Unix.fork() with
114       0 ->
115         begin
116           let lo=i*chunksize in
117           let hi=if i=ncores-1 then ln-1 else (i+1)*chunksize-1 in
118           let exc_handler e j = (* handle an exception at index j *)
119             info "error at index j=%d in (%d,%d), chunksize=%d of a total of %d got exception
120               j lo hi chunksize (hi-lo+1) (Printexc.to_string e) i;
121             exit 1
122           in
123           let v = compute al lo hi opid exc_handler in
124           marshal fdarr.(i) v;
125           exit 0
126         end
127     | -1 -> info "fork error: pid %d; i=%d" (Unix.getpid()) i;
128     | pid -> ()
129   done;
130   (* wait for all children *)
131   for i = 0 to ncores-1 do
132     try ignore(Unix.wait())
133     with Unix.Unix_error (Unix.ECHILD, _, _) -> ()
134   done;
135   (* read in all data *)
136   let res = ref [] in
137   (* iterate in reverse order, to accumulate in the right order *)
138   for i = 0 to ncores-1 do
139       res:= ((unmarshal fdarr.((ncores-1)-i)):'d)::!res;
140   done;
141   (* collect all results *)
142   collect !res
143 ;;
```

swh:1:cnt:d5214ff9562a1fe78db51944506ba48c20de3379;

origin=https://gitorious.org/parmap/parmap.git;lines=101-143

A solution to address the reproducibility crisis

McBane. 2020

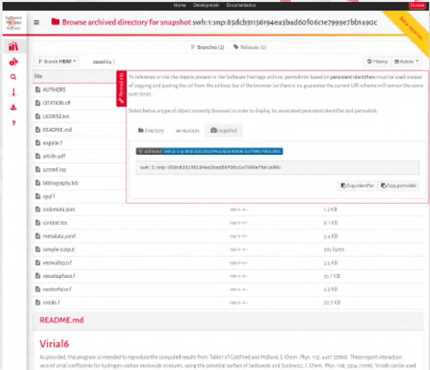


Copyright © 2020 G.C. McBane, released under a Creative Commons Attribution 4.0 International license.
Correspondence should be addressed to George C. McBane (mcbaneg@gvsu.edu)
The authors have declared that no competing interests exists.
Code is available at http://faculty.gvsu.edu/mcbaneg/virial6-resciencec.zip. – SWH swh:1:snp:85dcb31156194ea3bad60f06c1e7999e7bb1a90c.

ReScience C 6.1 (#1) – McBane 2020 1

George C. McBane. (2020). [Rp] Reproduction of interaction second virial coefficient calculation for H\$$_2$\$–CO interactions [J. Chem. Phys. vol. 112, 4417 (2000)]. Rescience C, 6(1), #1.

http://doi.org/10.5281/zenodo.3630224



Browse archived directory for snapshot swh:1:snp:85dcb31156194ea3bad60f06c1e7999e7bb1a90c

To reference or cite the objects present in the Software Heritage archive, permalinks based on **persistent identifiers** must be used instead of copying and pasting the url from the address bar of the browser (as there is no guarantee the current URI scheme will remain the same over time).

Select below a type of object currently browsed in order to display its associated persistent identifier and permalink.

swh:1:snp:85dcb31156194ea3bad60f06c1e7999e7bb1a90c

File		
AUTHORS		
CITATION.cff		
LICENSE.txt		
README.md		
angular.f		
article.pdf		
azoosf.inp		
bibliography.bib		
ogqf.f		
codemeta.json	-rw-r--r--	1.2 KB
content.tex	-rw-r--r--	9.1 KB
metadata.yaml	-rw-r--r--	3.4 KB
sample.output	-rw-r--r--	591 bytes
vecewalh2co.f	-rw-r--r--	3.5 KB
vecsetuph2co.f	-rw-r--r--	35.7 KB
vectorh2co.f	-rw-r--r--	2.3 KB
virials.f	-rw-r--r--	20.7 KB

README.md

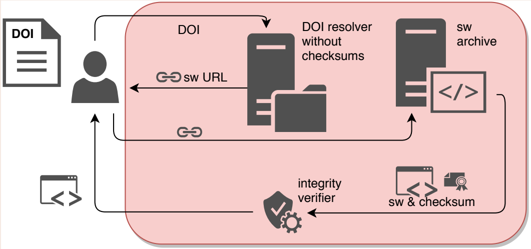
Virial6

As provided, this program is intended to reproduce the computed results from Table I of Gottfried and McBane, J. Chem. Phys. 112, 4417 (2000). These report interaction second virial coefficients for hydrogen-carbon monoxide mixtures, using the potential surface of Jankowski and Szalewicz, J. Chem. Phys. 108, 3554 (1998). Virial6 can be used

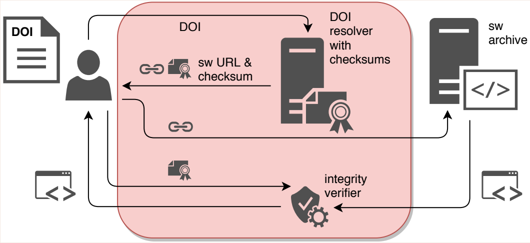
swh:1:snp:85dcb31156194ea3bad60f06c1e7999e7bb1a90c

Zoom on the trust model for identifiers

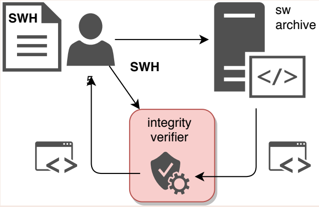
Trust model for usual DOIs



Trust model for DOIs with checksums



Trust model for SWH-IDs



Outline

1. Introduction
2. Academia's evolving practice
3. Archiving and referencing *all* the source code: Software Heritage
4. Zoom on the SWH-ID
5. Practical guidelines for archiving and referencing
6. What about metadata and citation?

Prepare your software source code complete guidelines

Prepare your public repository with:

- README, LICENSE, AUTHORS & codemeta.json files

What's a good README

extracted from Eric Steven Raymond and Make a README

MUST include:

- Name and a description of the software.

SHOULD include:

- how to run and use the source code
- build environment, installation, requirements

CAN include:

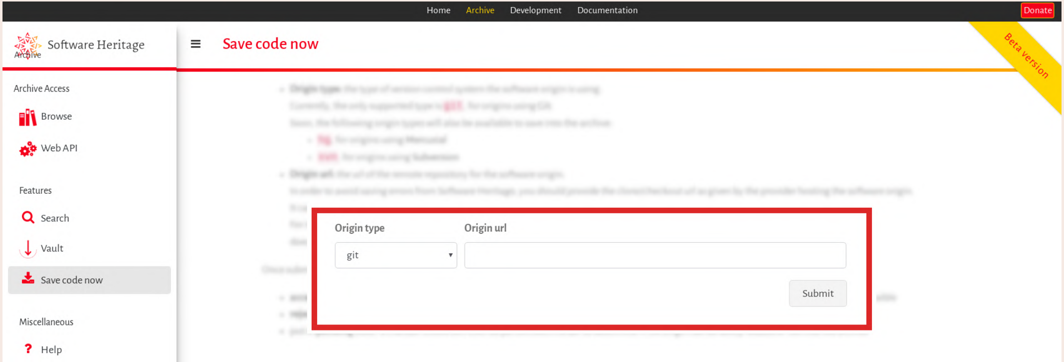
- project website or documentation pointer and recent news
- visuals

Submit save request on SWH

complete guidelines

Save code now on https://archive.softwareheritage.org/save/

- git, svn or mercurial
- intrinsic metadata files
- complete history



Save code now

- Origin type: the type of version control system the software origin is using.
  Currently, the only supported type is git, for origins using Git.
  Soon, the following origin types will also be available to save into the archive:
  - hg, for origins using Mercurial
  - svn, for origins using Subversion
- Origin url: the url of the remote repository for the software origin.
  In order to avoid saving errors from Software Heritage, you should provide the clone/checkout url as given by the provider hosting the software origin.

Origin type: git ▾

Origin url:

Reference software artifacts in your articles

complete guidelines

Choose the granularity level for the reference:

file (with code fragment)

swh:1:**cnt**:c60366bc03936eede6509b23307321faf1035e23;lines=473-537

... and add *;origin=https://github.com/sagemath/sage/*

James McCaffrey's **algorithm** in sageMath

directory

swh:1:**dir**:c6f07c2173a458d098de45d4c459a8f1916d900f

... and add *;origin=https://github.com/id-Software/Quake-III-Arena/*

source code of **Quake-III Arena** from id-Software

Reference software artifacts in your articles complete guidelines

specific release

swh:1:**rel**:22ece559cc7cc2364edc5e5593d63ae8bd229f9f
... and add *;origin=https://github.com/darktable-org/darktable/*

release 2.3.0 of Darktable, dated 24 December 2016

full snapshot (including all branches and all releases)

swh:1:**snp**:c7c108084bc0bf3d81436bf980b46e98bd338453
... and add *;origin=https://github.com/darktable-org/darktable/*

a **snapshot** of the entire Darktable repository (4 May 2017, GitHub)

Outline

1. Introduction
2. Academia's evolving practice
3. Archiving and referencing *all* the source code: Software Heritage
4. Zoom on the SWH-ID
5. Practical guidelines for archiving and referencing
6. **What about metadata and citation?**

It’s more complex than it seems!

Software is complex

Structure monolithic/composite; self-contained/external dependencies

Lifetime one-shot/long term

Community one man/one team/distributed community

Authorship complex set of roles

Authority institutions/organizations/communities/single person

Various granularities

Exact status of the source code for reproducibility, e.g.

“you can find at swh:1:cnt:cdf19c4487c43c76f3612557d4dc61f9131790a4;lines=146-187 the core algorithm used in this article”

(Major) release *“This functionality is available in OCaml version 4”*

Project *“Inria has created OCaml and Scikit-Learn”.*

Proposals for metadata and citation in the scholarly world

Refined ontology for contributors

- Design, Architecture,
- Coding, Testing, Debugging,
- Documentation, Maintenance, Support,
- Management

Reference is distinct from citation

- Reference is for *reproducibility*
  and now we can get it right!
- Citation is for *credit*
  and the jury is still out...

They must not be conflated

Keep the human in the loop

When *credit* is at stake, automation/crowdsourcing is not enough!
Humans *are needed* to get *quality information*

Experiments are ongoing on *moderated* software deposit ... (IDCC 2020)

Curated Archiving of Research Software Artifacts : lessons learned from the French open archive (HAL) https://hal.archives-ouvertes.fr/hal-02475835v1

Conclusion

Research software

- pillar of open science
- finally in the limelight

Doing it right is not easy

- *simplistic* approaches, "just data", ...
- soon part of *research evaluation*

You can help make a change

- leverage Software Heritage in conferences and journals for *archival* and *reference*
- join the conversation on *software citation* and *software evaluation* criteria

Where can you participate?

- Software Source Code Interest group - RDA-SSC IG
- Software Source code Identification Working Group - RDA-Force11-SCID WG
- Software Citation Implementation Working Group - Force11-SCIWG

Come in, we're open !

`www.softwareheritage.org` — learn more
`save.softwareheritage.org` — save code now
`www.softwareheritage.org/swhap` — legacy software acquisition process
`forge.softwareheritage.org` — our own code

Questions?

References

Jean-François Abramatic, Roberto Di Cosmo, Stefano Zacchiroli
Building the Universal Archive of Source Code, CACM, October 2018 (10.1145/3183558)

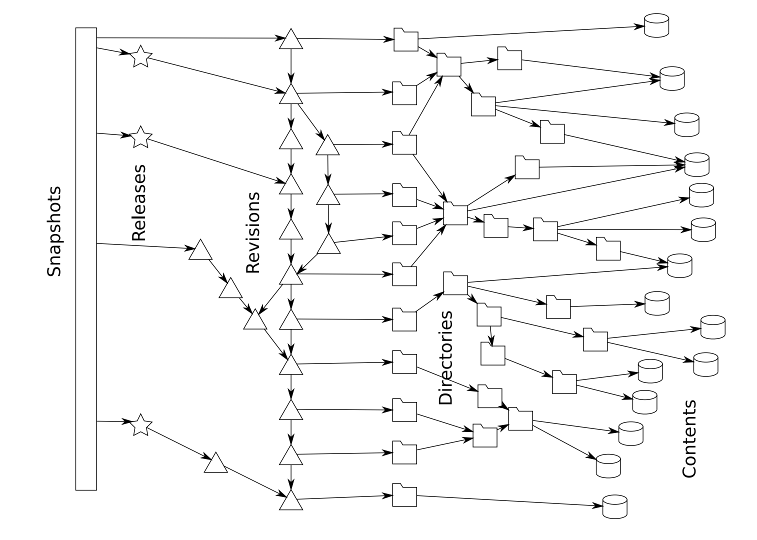
Roberto Di Cosmo, Morane Gruenpeter, Stefano Zacchiroli
Referencing Source Code Artifacts: a Separate Concern in Software Citation,
CiSE 2020 (10.1109/MCSE.2019.2963148) (hal-02446202)

Pierre Alliez, Roberto Di Cosmo, Benjamin Guedj, Alain Girault, Mohand-Said Hacid, Arnaud Legrand and Nicolas Rougier
Attributing and referencing (research) software: Best practices and outlook from Inria,
CiSE 2020 (10.1109/MCSE.2019.2949413) (hal-02135891)

Outline

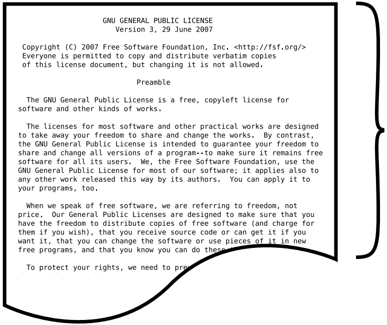
7. Worked example Merkle tree

A worked example



A worked example

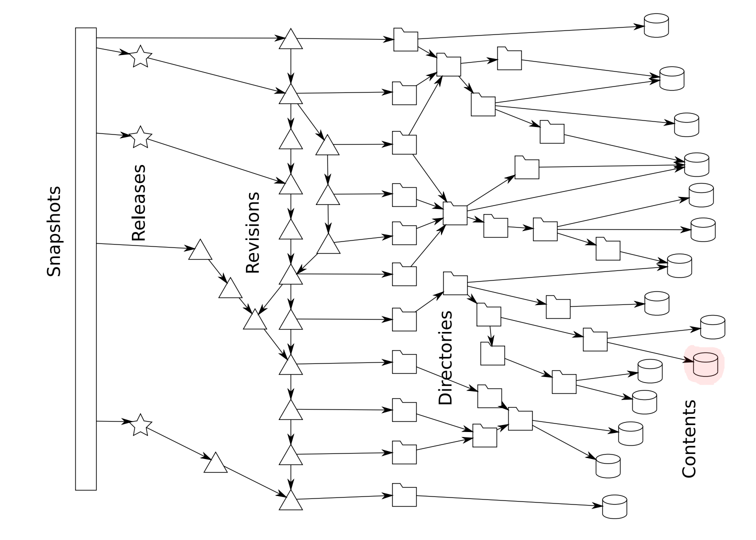
Contents



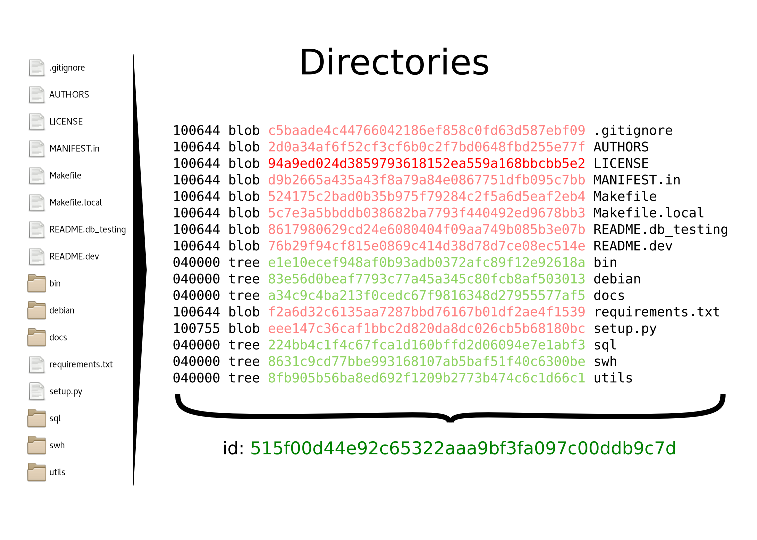
```
                    GNU GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

 Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

                            Preamble

  The GNU General Public License is a free, copyleft license for
software and other kinds of works.

  The licenses for most software and other practical works are designed
to take away your freedom to share and change the works.  By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users.  We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors.  You can apply it to
your programs, too.

  When we speak of free software, we are referring to freedom, not
price.  Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these

  To protect your rights, we need to pre
```

sha1: 8624bcdae55baeef...
sha256: 8ceb4b9ee5aded...
sha1_git: 94a9ed024d385...
length: 35147

A worked example



A worked example

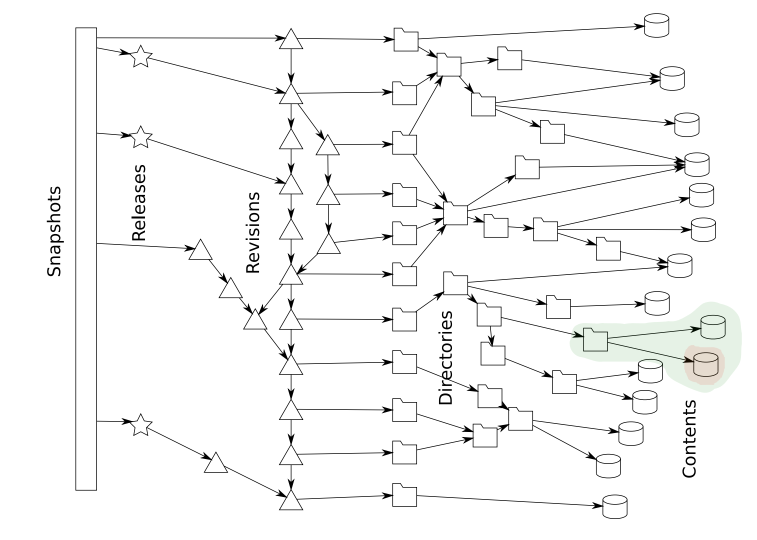


Directories

```
100644 blob c5baade4c44766042186ef858c0fd63d587ebf09 .gitignore
100644 blob 2d0a34af6f52cf3cf6b0c2f7bd0648fbd255e77f AUTHORS
100644 blob 94a9ed024d3859793618152ea559a168bbcbb5e2 LICENSE
100644 blob d9b2665a435a43f8a79a84e0867751dfb095c7bb MANIFEST.in
100644 blob 524175c2bad0b35b975f79284c2f5a6d5eaf2eb4 Makefile
100644 blob 5c7e3a5bbddb038682ba7793f440492ed9678bb3 Makefile.local
100644 blob 8617980629cd24e6080404f09aa749b085b3e07b README.db_testing
100644 blob 76b29f94cf815e0869c414d38d78d7ce08ec514e README.dev
040000 tree e1e10ecef948af0b93adb0372afc89f12e92618a bin
040000 tree 83e56d0beaf7793c77a45a345c80fcb8af503013 debian
040000 tree a34c9c4ba213f0cedc67f9816348d27955577af5 docs
100644 blob f2a6d32c6135aa7287bbd76167b01df2ae4f1539 requirements.txt
100755 blob eee147c36caf1bbc2d820da8dc026cb5b68180bc setup.py
040000 tree 224bb4c1f4c67fca1d160bffd2d06094e7e1abf3 sql
040000 tree 8631c9cd77bbe993168107ab5baf51f40c6300be swh
040000 tree 8fb905b56ba8ed692f1209b2773b474c6c1d66c1 utils
```

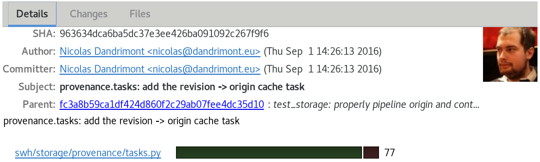
id: 515f00d44e92c65322aaa9bf3fa097c00ddb9c7d

A worked example



A worked example

Revisions



SHA: 963634dca6ba5dc37e3ee426ba091092c267f9f6

Author: Nicolas Dandrimont <nicolas@dandrimont.eu> (Thu Sep 1 14:26:13 2016)

Committer: Nicolas Dandrimont <nicolas@dandrimont.eu> (Thu Sep 1 14:26:13 2016)

Subject: **provenance.tasks: add the revision -> origin cache task**

Parent: fc3a8b59ca1df424d860f2c29ab07fee4dc35d10 : *test_storage: properly pipeline origin and cont...*

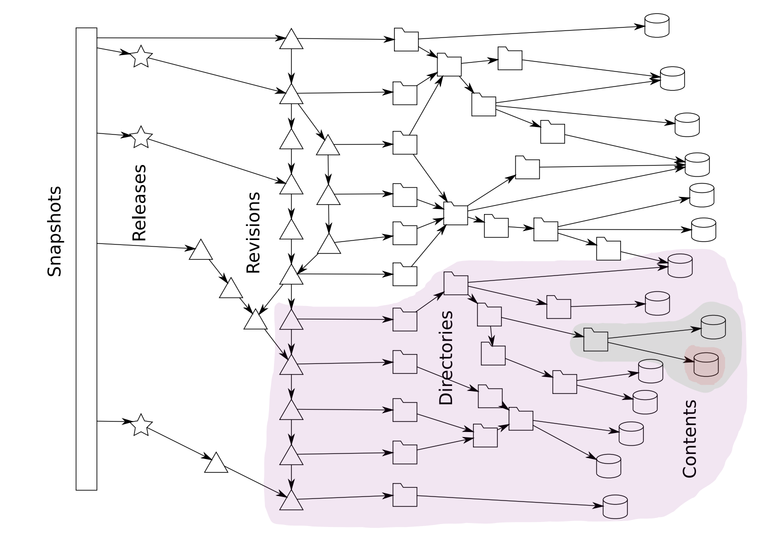
provenance.tasks: add the revision -> origin cache task

swh/storage/provenance/tasks.py 77

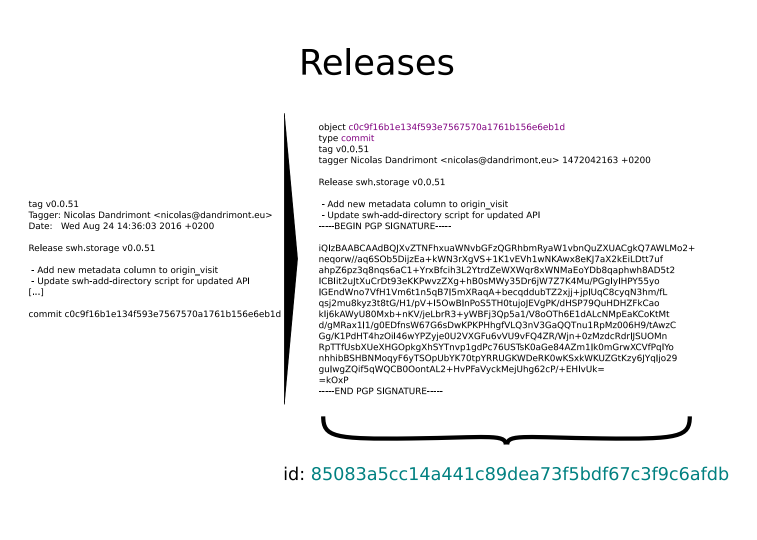
```
tree 515f00d44e92c65322aaa9bf3fa097c00ddb9c7d
parent fc3a8b59ca1df424d860f2c29ab07fee4dc35d10
author Nicolas Dandrimont <nicolas@dandrimont.eu> 1472732773 +0200
committer Nicolas Dandrimont <nicolas@dandrimont.eu> 1472732773 +0200

provenance.tasks: add the revision -> origin cache task
```

id: 963634dca6ba5dc37e3ee426ba091092c267f9f6

A worked example



A worked example



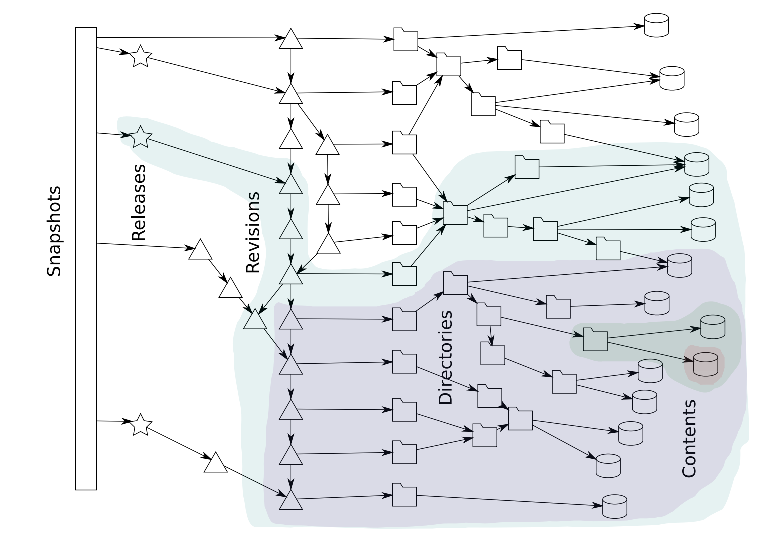
Releases

```
tag v0.0.51
Tagger: Nicolas Dandrimont <nicolas@dandrimont.eu>
Date:   Wed Aug 24 14:36:03 2016 +0200

Release swh.storage v0.0.51

 - Add new metadata column to origin_visit
 - Update swh-add-directory script for updated API
[...]

commit c0c9f16b1e134f593e7567570a1761b156e6eb1d
```

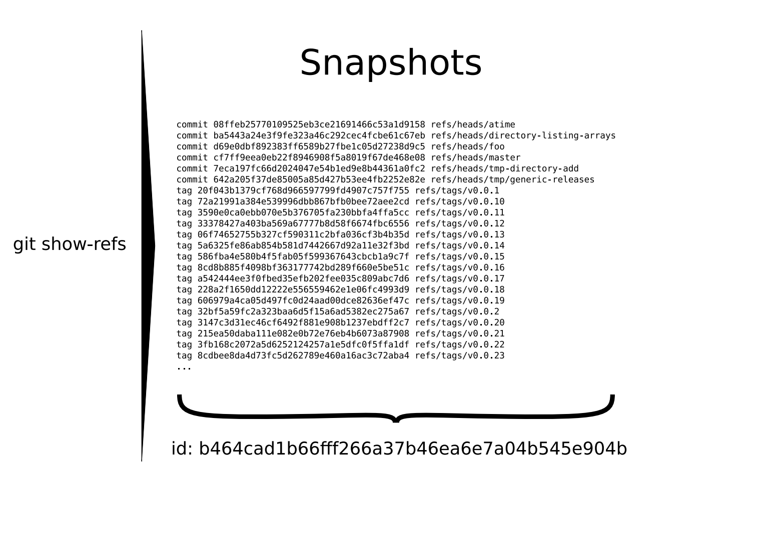
```
object c0c9f16b1e134f593e7567570a1761b156e6eb1d
type commit
tag v0.0.51
tagger Nicolas Dandrimont <nicolas@dandrimont.eu> 1472042163 +0200

Release swh.storage v0.0.51

 - Add new metadata column to origin_visit
 - Update swh-add-directory script for updated API
-----BEGIN PGP SIGNATURE-----

iQIzBAABCAAdBQJXvZTNFhxuaWNvbGFzQGRhbmRyaW1vbnQuZXUACgkQ7AWLMo2+
neqorw//aq6SOb5DijzEa+kWN3rXgVS+1K1vEVh1wNKAwx8eKJ7aX2kEiLDtt7uf
ahpZ6pz3q8nqs6aC1+YrxBfcih3L2YtrdZeWXWqr8xWNMaEoYDb8qaphwh8AD5t2
lCBIit2uJtXuCrDt93eKKPwvzZXg+hB0sMWy35Dr6jW7Z7K4Mu/PGgIyIHPY55yo
lGEndWno7VfH1Vm6t1n5qB7I5mXRaqA+becqddubTZ2xjj+jpIUqC8cyqN3hm/fL
qsj2mu8kyz3t8tG/H1/pV+I5OwBInPoS5TH0tujoJEVgPK/dHSP79QuHDHZFkCao
kIj6kAWyU80Mxb+nKV/jeLbrR3+yWBFj3Qp5a1/V8oOTh6E1dALcNMpEaKCoKtMt
d/gMRax1I1/g0EDfnsW67G6sDwKPKPHhgfVLQ3nV3GaQQTnu1RpMz006H9/tAwzC
Gg/K1PdHT4hzOiI46wYPZyje0U2VXGFu6vVU9vFQ4ZR/Wjn+0zMzdcRdrIJSUOMn
RpTTfUsbXUeXHGOpkgXhSYTnvp1gdPc76USTsK0aGe84AZm1Ik0mGrwXCVfPqIYo
nhhibBSHBNMoqyF6yTSOpUbYK70tpYRRUGKWDeRK0wKSxkWKUZGtKzy6JYqIjo29
guIwgZQif5qWQCB0OontAL2+HvPFaVyckMejUhg62cP/+EHIvUk=
=kOxP
-----END PGP SIGNATURE-----
```

id: 85083a5cc14a441c89dea73f5bdf67c3f9c6afdb

A worked example



A worked example



Snapshots

git show-refs

```
commit 08ffeb25770109525eb3ce21691466c53a1d9158 refs/heads/atime
commit ba5443a24e3f9fe323a46c292cec4fcbe61c67eb refs/heads/directory-listing-arrays
commit d69e0dbf892383ff6589b27fbe1c05d27238d9c5 refs/heads/foo
commit cf7ff9eea0eb22f8946908f5a8019f67de468e08 refs/heads/master
commit 7eca197fc66d2024047e54b1ed9e8b44361a0fc2 refs/heads/tmp-directory-add
commit 642a205f37de85005a85d427b53ee4fb2252e82e refs/heads/tmp/generic-releases
tag 20f043b1379cf768d966597799fd4907c757f755 refs/tags/v0.0.1
tag 72a21991a384e539996dbb867bfb0bee72aee2cd refs/tags/v0.0.10
tag 3590e0ca0ebb070e5b376705fa230bbfa4ffa5cc refs/tags/v0.0.11
tag 33378427a403ba569a67777b8d58f6674fbc6556 refs/tags/v0.0.12
tag 06f74652755b327cf590311c2bfa036cf3b4b35d refs/tags/v0.0.13
tag 5a6325fe86ab854b581d7442667d92a11e32f3bd refs/tags/v0.0.14
tag 586fba4e580b4f5fab05f599367643cbcb1a9c7f refs/tags/v0.0.15
tag 8cd8b885f4098bf363177742bd289f660e5be51c refs/tags/v0.0.16
tag a542444ee3f0fbed35efb202fee035c809abc7d6 refs/tags/v0.0.17
tag 228a2f1650dd12222e556559462e1e06fc4993d9 refs/tags/v0.0.18
tag 606979a4ca05d497fc0d24aad00dce82636ef47c refs/tags/v0.0.19
tag 32bf5a59fc2a323baa6d5f15a6ad5382ec275a67 refs/tags/v0.0.2
tag 3147c3d31ec46cf6492f881e908b1237ebdff2c7 refs/tags/v0.0.20
tag 215ea50daba111e082e0b72e76eb4b6073a87908 refs/tags/v0.0.21
tag 3fb168c2072a5d6252124257a1e5dfc0f5ffa1df refs/tags/v0.0.22
tag 8cdbee8da4d73fc5d262789e460a16ac3c72aba4 refs/tags/v0.0.23
...
```

id: b464cad1b66fff266a37b46ea6e7a04b545e904b