Adding third party notices to ChromeDriver

This Document is Public

Authors: nechaev@chromium.org
September 2022

Short link: go/chormedriver-license

One-page overview

Summary

Amount of licenses in third party dependencies of ChromeDriver is estimated as 35+. None of them are shipped with the binaries even though the 'notice' licenses always have to be shipped. We also have to provide the user with the license of the ChromeDriver itself.
In this document we consider how to do this.

Platforms

Mac, Windows, Linux

Team

nechaev@chromium.org

Bug

Chromedriver:4203 "Improve credits information"

Code affected

Scripts and configurations related to ChromeDriver build and release process

---

Signed off by

Name	Write (not) LGTM in this column
mathias@chromium.org	LGTM
TODO: Any maintainer of licenses.py?	

Core user stories

User story 1: User downloads ChromeDriver from the Downloads page

User navigates to the Downloads page and downloads the desired ChromeDriver version. The license and the third party notices must be downloaded together with ChromeDriver binaries.

User story 2: User downloads ChromeDriver Canary

User navigates to the Canary page for Linux or another platform and downloads the desired ChromeDriver build.
The license and the third party notices must be downloaded together with ChromeDriver binaries.

User story 3: Developer cuts out ChromeDriver manually

Developer is working on ChromeDriver version X and needs to cut out a new release of ChromeDriver version Y manually. The developer is able to do that without switching any branches or rebuilding ChromeDriver.

User story 4: Client software is able to see the license and the third party notices

If user software depending on ChromeDriver needs the licenses of its dependencies it would be good if we provided the license and the notices in some friendly way.

Assumptions

- All third party dependencies of ChromeDriver have Chromium compatible licenses. This is ensured by the third party review process.
- We assume that the credits information provided by Chrome also mentions the patches applied to the third party dependencies where it is appropriate.

Design

ChromeDriver builds are published to the Downloads page manually with help of the scripts in the //chrome/test/chromedriver/support/release directory. Adding a step for the third party notice generation in these scripts does not satisfy the requirement that this information must be available in the canary builds. The license and the notices must be readily available during the packaging process when the ChromeDriver binary is compressed into a ZIP archive.

How to provide the third party notices

Plain concatenation of ChromeDrive license and third party licenses in the LICENSE file is inappropriate as it will not be obvious to the user that one of them belongs to the ChromeDriver while the others belong to the dependencies.

Providing the ChromeDriver license in the beginning of the LICENSE file and then giving a list of components and their licenses seems to be a more appropriate solution. This approach is used in some other chromium family products (cronet, fichsia/cipd, ios/web_view) and looks to be acceptable.

Providing the ChromeDriver license in the LICENSE file and the third party notices in the THIRD_PARTY_NOTICES.txt might be a little bit clearer to the user. This practice is also mentioned in some google documents.

How do we discover the third party notices

There is a software //tools/licenses.py that generates the third party notices for the Chrome browser. It has a mode in which it accepts a GN target, discovers all its dependencies and concatenates their licenses into a single LICENSE file of form:

<Chromium License Text>
[
========

<Dependency name>
========
<Dependency license text>
]

This works pretty well excepting that it does not discover and mention the following dependencies:

- //third_party/bidimapper
- //third_party/webdriver (aka selenium-atoms)
- //third_party/webdriver dependencies

The first two are skipped because they do not create any targets. They just contain some files included directly by ChromeDriver's BUILD.gn (selenium-atoms) or used there in order to generate some code (bidimapper).
The third one is not discovered because these dependencies are not accessible from the source code tree in principle. They are checked out while we are updating the atoms and then erased again. See the README.chromium for more details.

To make the first two dependencies discoverable we can create BUILD.gn files under the //third_party/webdriver and //third_party/bidimapper folders. The former can declare a source_set target containing atoms.h and atoms.cc files. The latter can declare a 'copy' target to export its mapper.js file.
Making the dependencies of //third_party/webdriver discoverable is more complicated. We considered the following SA (selenium-atoms) alternatives here.

Alternative SA1: Selenium becomes a submodule of Chromium.

With this alternative the dependencies become reachable from the Chromium source tree. The license generating tool //tools/licenses.py does not work well with the nested dependencies under the //third_party directory. It will require some modifications.
Pros:

- If dependencies of selenium atoms are changed these changes will be discovered automatically

Cons:

- The checkout size of Chromium source code grows significantly as Selenium is quite big (~6K files, ~9M LOC)
- The ChromeDriver build time will increase.
- The //tools/licenses.py requires some modifications that can affect its work with other dependencies.

The checkout size penalty looks to be too heavy for the small portion of the code that we really use.

Alternative SA2: Fake targets

We can create a folder for each selenium-atoms dependency somewhere. This folder will contain the corresponding LICENSE file and a BUILD.gn file with an empty source_set target. The dependency on such an empty source_set can be added either to the BUILD.gn of selenium-atoms or to the BUILD.gn of the ChromeDriver itself.
We consider the following possible locations where to place these new folders:

- //third_party/webdriver. It turned out that //tools/licenses.py is not able to discover any nested dependencies under the //third_party directory. This approach will require significant modifications in the //tools/licenses.py tool. This is error prone therefore we will reject this option.
- //chrome/test/chromedriver/third_party. This directory was added to the root .gitignore list in 2019 and is missing now. Having a third_party directory under //chrome/test/chromedriver might create temptation for the future developers to add real third party dependencies there while this practice is discouraged. We want to avoid any misunderstandings in the future and therefore we will avoid this solution.
- //third_party. With this approach we have the following question: “What if somebody needs to add one of these dependencies in the future for real?”. In such a situation the target can become something different and in particular it can push some new and unused code or resources into the ChromeDriver.

Pros

- The solution mostly leverages the existing target discovery mechanism

Cons

- The location for fake targets is not easy to choose. There are different inconveniences related to each location.
- Every time the selenium-atoms are updated the corresponding fake targets need to be manually checked and, if needed, updated as well.
- Some locations do not have ‘reference locality’ property - the licenses are far from the atoms in the source tree.

The alternative with fake targets looks appealing but it is not quite clear how to implement it well. Therefore we reluctantly reject it.

Alternative SA3: Special cases

The //tools/licenses.py tool supports special cases. One can create a record, analogical to the contents of the README.chromium, that has a fictive component path and the ‘License File’ field referring to the license file located somewhere in the source tree. This is how the licenses of ‘strongtalk’ and ‘fdlibm’ are included by v8.
We can do the following:

- Create a special case for each dependency of selenium atoms.

- Create a license file for each selenium dependency directly under the //third_party/webdriver directory. The license file names will have "LICENSE.<dependency_name>" name structure by analogy with 'v8' dependencies.
- Pass the fake component path of each dependency via the existing argument '--extra-third-party-dirs' to the //tools/licenses.py license_file command.

The resulting LICENSE will contain all third party dependencies + licenses of all special case selenium dependencies.
As //tools/licenses.py does not support the '--extra-third-party-dirs' parameter in its GN target mode we will need to add such support there.
Pros:

- Easy to implement
- Virtual dependencies not used during the ChromeDriver build process are explicitly declared as special cases - cognitive ease for the maintainer.
- Reference locality - the licenses are not far from the selenium atoms.

Cons:

- Every time the selenium-atoms are updated the corresponding licenses need to be manually synchronized with selenium atoms dependencies.

This alternative looks to be winning mostly due to its cognitive ease qualities for the maintainer.

Open Issues

It is not clear how we integrate license generation into the packaging process so that the canary, dev, beta and stable builds had the licensing information and third party notices alongside the binary in the archive.

Alternatives Considered

Alternative #1. Chrome credits + ChromeDriver extras

In this approach THIRD_PARTY_NOTICES.txt mentions that the most of notices can be found at chrome://credits. The few ones not mentioned there are listed in the THIRD_PARTY_NOTICES.txt itself.
The third party notice file can be created manually and stored under the source control as it is relatively small. The packaging tool can bundle it together with the ChromeDriver binary and the LICENSE file in a single ZIP archive.

Pros:

- Easy to implement

Cons:

- Many dependencies shown in chrome://credits are not used by ChromeDriver.

- As ChromeDriver is distributed separately from Chrome we still do not provide the majority of notices together with the binary.
- ChromeDriver is able to work with the (Major+1) version of Chrome that might have different dependencies. Actually we might expect that even different minor versions can have different third party dependencies.
- Some third party notices must be updated manually

Prototype: CL (with some deviations)

Alternative #2. Dependency discovery + virtual targets

The license is generated by the packaging process with help of //tools/licenses.py that automatically discovers most of the dependencies. Few dependencies that cannot be automatically discovered are handled by //tools/licenses.py as special cases - we need to feed them via the '--extra-third-party-dirs' parameter.
The tool generates a single LICENSE file with Chromium license in the beginning and third party licenses listed below after the name of each third party dependency.
This is the script that will be executed by the packaging process behind the scenes:

```
tools/licenses.py license_file '<path-to-chromedriver-build-dir>/LICENSE'
--gn-target '//chrome/test/chromedriver:chromedriver' --extra-third-party-dirs
'["path-to-virtual-dependency1", "path-to-virtual-dependency2", ...]' --gn-out-dir
out/Debug
```

Pros:
- Majority of third party notices are actualized automatically
- All third party notices are always distributed alongside the binary

Cons:
- Some third party notices must be updated manually

Prototype: CL

Metrics

Success metrics

All the above mentioned user stories are covered.

Regression metrics

The only possible regression is if licensing or credits information for unrelated products become incorrect. Visual check should be sufficient to see that nothing was broken by the change in //tools/licenses.py.

Rollout plan

Waterfall

Core principle considerations

Speed

Chrome or ChromeDriver performance is unaffected.
The packaging job on the build bot will require appx. 15 seconds in order to generate the LICENSE file.

Security

No security risks.

Privacy considerations

Privacy is not affected.

Testing plan

We need to extend the //tools/licenses.py tests with the test case covering the changes related to the ‘ --extra-third-party-dirs’ parameter in the GN target mode.

Remarks

Amount of ChromeDriver third party dependencies assessment

In order to estimate the amount of LICENSE files in the third party dependencies of ChromeDriver we used the following script:

```
for dir in $(gn desc out/Debug 'chrome/test/chromedriver:chromedriver' deps --all |
grep -E '\/\/third_party' | cut -d':' -f1 | uniq); do path="${dir#//}/LICENSE"; [
-f "$path" ] && echo "$path"; done | wc -l
```

The result is not precise as it does not find //third_party/bidimapper, //third_party/webdriver and its dependencies. It possibly does not find some other dependencies as well.

TODO

- Update Readme.chromium of selenium-atoms: mention there that the licenses need to be updated manually every time we update the atoms. Also explain there how special cases are implemented.
- Mention that selenium.atoms is patched and explain how in its license

Followup work

How will you assess the success of this work? What needs to be followed-up on? What (experimental code, for example) needs to be cleaned up after the code has reached the stable channel?