Using AboutCode Toolkit to Document Your Software Assets

AboutCode Toolkit Defined

AboutCode Toolkit is a tool for your software development team to document your code inside your codebase, typically in preparation for a product release, side-by-side with the actual code. AboutCode Toolkit files have a simple, standard format that identifies components and their associated licenses. The current AboutCode Toolkit subcommands are:

- **attrib**: Generate a Product Attribution notice document (HTML format) from your AboutCode Toolkit files. You can also generate documents for other purposes (such as a License Reference) by varying your input control file and your template.
- **check**: A simple command to validate the AboutCode Toolkit files and output errors/warnings if any on the terminal.
- **collect_redist_src**: A command to collect and copy sources that have 'redistribute' flagged as 'True' in AboutCode Toolkit files or from an inventory.
- **gen**: Create AboutCode Toolkit files from a Software Inventory file (.csv or .json format) which is typically created from a software audit, and insert these AboutCode Toolkit files into your codebase. You can regenerate the AboutCode Toolkit files from a new Software Inventory file whenever you make changes.
- **inventory**: Generate a Software Inventory list (.csv or .json format) from your codebase based on your AboutCode Toolkit files. Note that this Software Inventory will only include components that have AboutCode Toolkit data. In another word, if you do not create AboutCode Toolkit files for your own original software components, these components will not show up in the generated inventory.
- **transform**: A command to transform an input CSV/JSON by applying renaming and/or filtering and then output to a new CSV/JSON file.

Additional AboutCode Toolkit information is available at:

- http://www.aboutcode.org/ for an overview and a link to the ABOUT File specification.
- https://github.com/nexB/aboutcode-toolkit/ for the AboutCode Toolkit tools.

Key Terminology

Some key terminology that applies to AboutCode Toolkit tool usage:

- **Software Inventory or Inventory** - means a list of all of the components in a Development codebase and the associated data about those components with a focus on software pedigree/provenance- related data for open source and third-party components.
- **Product BOM or BOM** - means a subset list of the components in a Development codebase (Software Inventory) that are Deployed on a particular Product Release (a Product Bill of Materials).

Using gen to Generate AboutCode Toolkit Files

Prepare Your Software Inventory for gen Standard Field Names

You should start with a software inventory of your codebase in spreadsheet or JSON format. You need to prepare a version of it that will identify the field values that you want to appear in your .ABOUT files. Note the following standard field names (defined in the ABOUT File Specification), which gen will use to look for the values that it will store in your generated .ABOUT files, as well as any additional text files that you identify, which it will copy and store next to the .ABOUT files.

Standard Field Name	Description	Notes
about_resource	Name/path of the component resource	Mandatory.
name	Component name	Mandatory
version	Component version	Optional
download_url	Direct URL to download the original file or archive documented by this ABOUT file	Optional

description	Component description	Optional
homepage_url	URL to the homepage for this component	Optional
package_url	Package URL for this component (See https://github.com/package-url/purl-spec for SPEC)	Optional
notes	notes text	Optional
license_expression	Expression for the license of the component using DejaCode Enterprise license key(s).	Optional. You can separate each identifier using " or " and " and " to document the relationship between multiple license identifiers, such as a choice among multiple licenses.
license_key	DejaCode Enterprise license key for the component.	Optional. gen will obtain license information from DejaCode Enterprise if the --fetch-license option is set, including the license text, in order to create and write the appropriate .LICENSE file in the .ABOUT file target directory.
license_name	License name for the component.	Optional. This field will be generated if the --fetch-license option is set.
license file	license file name	Optional. gen will look for the file name (if a directory is specified in the --reference option) to copy that file to the .ABOUT file target directory.
license_url	URL to the license text for the component	Optional
copyright	copyright statement for the component	Optional
notice_file	notice text file name	Optional
notice_url	URL to the notice text for the component	Optional
redistribute	Yes/No. Does the component license require source redistribution.	Optional
attribute	Yes/No. Does the component license require publishing an attribution or credit notice.	Optional
track_changes	Yes/No. Does the component license require tracking changes made to the component.	Optional
modified	Yes/No. Have the component been modified.	Optional
internal_use_only	Yes/No. Is the component internal use only.	Optional
changelog_file	changelog text file name	Optional
owner	name of the organization or person that owns or provides the component	Optional
owner_url	URL to the owner for the component	Optional
contact	Contact information	Optional
author	author of the component	Optional
author_file	author text file name	Optional

vcs_tool	Name of the version control tool.	Optional
vcs_repository	Name of the version control repository.	Optional
vcs_path	Name of the version control path.	Optional
vcs_tag	Name of the version control tag.	Optional
vcs_branch	Name of the version control branch.	Optional
vcs_revision	Name of the version control revision.	Optional
checksum_md5	MD5 value for the file	Optional
checksum_sha1	SHA1 value for the file	Optional
checksum_sha256	SHA256 value for the file	Optional
spec_version	The version of the ABOUT file format specification used for this file.	Optional

Fields Renaming and Optional Custom Fields

Since your input's field name may not match with the AboutCode Toolkit standard field name, you can use the transform subcommand to do the transformation.

A transform configuration file is used to describe which transformations and validations to apply to a source CSV/JSON file. This is a simple text file using YAML format, using the same format as an .ABOUT file.

The attributes that can be set in a configuration file are:

- field_renamings: An optional map of source field name to target new field name that is used to rename CSV/JSON fields.

For instance with this configuration, the field "Directory/Location" will be renamed to "about_resource" and "foo" to "bar":

```
field_renamings:
   about_resource : 'Directory/Location'
   bar : foo
```

The renaming is always applied first before other transforms and checks. All other field names referenced below are AFTER the renaming have been applied.

- required_fields: An optional list of required field names that must have a value, beyond the standard field names. If a source CSV/JSON does not have such a field or an entry is missing a value for a required field, an error is reported.

For instance with this configuration, an error will be reported if the fields "name" and "version" are missing, or if any entry does not have a value set for these fields:

```
required_fields:
   - name
   - version
```

- field_filters: An optional list of fields that should be kept in the transformed file. If this list is provided, only the fields that are in the list will be kept. All others will be filtered out even if they are AboutCode Toolkit standard fields. If this list is not provided, all source fields are kept in the transformed target file.

For instance with this configuration, the target file will only contains the "name" and "version" fields:

```
field_filters:
   - name
   - version
```

- exclude_fields: An optional list of field names that should be excluded in the transformed file. If this list is provided, all the fields from the source file that should be excluded in the target file must be listed. Excluding required fields will cause an error. If this list is not provided, all source fields are kept in the transformed target file.

For instance with this configuration, the target file will not contain the "type" and "temp" fields:

```
exclude_fields:
    - type
    - temp
```

Run gen to Generate AboutCode Toolkit Files

When your software inventory is ready, you can save it as a .csv or .json file, and use it as input to run gen to generate your AboutCode Toolkit files. The official gen parameters are defined here:

- https://github.com/nexB/aboutcode-toolkit/blob/develop/REFERENCE.rst

Here is an example of a gen command:

about gen --fetch-license {{your license library api}} {{your license library api key}} --reference /Users/harrypotter/myAboutFiles/ /Users/harrypotter/myAboutFiles/myProject-bom.csv /Users/harrypotter/myAboutFiles/

Note that this example gen command does the following:

- Activates the --fetch-license option to get license text.
- Activates the --reference option to get license text files and notice text files that you have specified in your software inventory to be copied next to the associated .ABOUT files when those are created.
- Specifies the path of the software inventory to control the processing.
- Specifies a target output directory.

Review your generated AboutCode Toolkit files to determine if they meet your requirements. Here is a simple example of a linux-redhat-7.2.ABOUT file that documents the directory /linux-redhat-7.2/ :

```
    about_resource: .
    name: Linux RedHat
    version: v 7.2
    attribute: Y
    copyright: Copyright (c) RedHat, Inc.
    license_expression: gpl-2.0
    licenses:
        -   key: gpl-2.0
            name: GNU General Public License 2.0
            file: gpl-2.0.LICENSE
    owner: Red Hat
    redistribute: Y
```

You can make the appropriate changes to your input software inventory and then run gen as often as necessary to replace the generated AboutCode Toolkit files with the improved output.

Using attrib to Generate a Product Attribution Notice Package

Prepare a Filtered Product BOM to Use as Input to attrib

The Software Inventory that you prepared for gen most likely includes components that do not need to appear in a product attribution notice package; for example:

- Components in your codebase that are not Deployed on the final product (e.g. build tools, testing tools, internal documentation).
- Components in your codebase under licenses that do not require attribution (e.g. proprietary packages, commercial products).

There are two options here:

- Edit the jinja2 template to only include the one that have value in attribute field such as:

  {% if about_object.attribute.value %} ...

- You should prepare a filtered version of your software inventory (the one that you used for gen) by removing the rows that identify components which should not be included in a product attribution notice package, and saving that filtered version as your Product BOM.

Prepare an Attribution Template to Use as Input to attrib

You can run attrib using the default_html.template (or default_json.template if want JSON output) provided with the AboutCode Toolkit tools:

https://github.com/nexB/aboutcode-toolkit/blob/develop/templates/default_html.template

If you choose to do that, you will most likely want to edit the generated .html file to provide header information about your own organization and product.

Running attrib with the default_html.template file is probably your best choice when you are still testing your AboutCode Toolkit process. Once you have a good understanding of the generated output, you can customize the template to provide the standard text that you want to see whenever you generate product attribution for your organization. You can also create alternative versions of the template to use attrib to generate other kinds of documents, such as a License Reference.

Use jinja2 Features to Customize Your Attribution Template

The attrib tool makes use of the open source python library **jinja2** (http://jinja.pocoo.org/docs/dev/templates/) in order to extend .html capabilities and transform AboutCode Toolkit input data into the final format of the generated attribution file. The **default_html.template **file contains text that complies with jinja2 syntax specifications in order to support grouping, ordering, formatting and presentation of your AboutCode Toolkit data. If your attribution requirements are complex, you may wish to study the jinja2 documentation to modify the default_html.template logic or create your own template; alternatively, here are a few relatively simple concepts that relate to the attribution document domain.

The simplest modifications to the default_html.template file involve the labels and standard text. For example, here is the default template text for the Table of Contents:

```
<div class="oss-table-of-contents">
    {% for about_object in abouts %}
        <p><a href="#component_{{ loop.index0 }}">{{ about_object.name.value }}
        {% if about_object.version.value %} {{ about_object.version.value }}
        {% endif %}</a></p>
    {% endfor %}
</div>
```

If you would prefer something other than a simple space between the component name and the component version, you can modify it to something like this:

```
<div class="oss-table-of-contents">
    {% for about_object in abouts %}
        <p><a href="#component_{{ loop.index0 }}">{{ about_object.name.value }}
        {% if about_object.version.value %}  - Version  {{ about_object.version.value }}
        {% endif %}</a></p>
    {% endfor %}
</div>
```

The "if about_object.version.value" is checking for a component version, and if one exists it generates output text that is either a space followed by the actual version value, or, as in this customized template, it generates output text as " - Version ", followed by the actual version value. You will, of course, want to test your output to get exactly the results that you need.

Note that you can actually use attrib to generate an AboutCode Toolkit-sourced document of any kind for varying business purposes, and you may want to change the grouping/ordering of the data for different reporting purposes. (Here we get into somewhat more complex usage of jinja2 features, and you may wish to consult the jinja2 documentation to reach a more comprehensive understanding of the syntax and features.) The default ordering is by component, but In the following example, which is intended to support a "license reference" rather than an attribution document, the customized template modifies the data grouping to use a custom field called "confirmed license":

```
<div class="oss-table-of-contents">
    {% for group in abouts | groupby('confirmed_license') %}
    <p>
        {% for license in group.grouper.value %}
        <a href="#group_{{ loop.index0 }}">{{ license }}
        </a>
        {% endfor %}
    </p>
    {% endfor %}
</div>
```

After the table of contents, this example customized template continues with the license details using the jinja2 for-loop capabilities. Notice that the variable "group.grouper.value" is actually the license name here, and that “License URL” can be any URL that you have chosen to store in your .ABOUT files:

```
{% for group in abouts | groupby('confirmed_license') %}
    {% for confirmed_license in group.grouper.value %}

    <div id="group_{{ loop.index0 }}">
    <h3>{{ confirmed_license }}</h3>
    <p>This product contains the following open source software packages licensed under the terms of the license: {{confirmed_li

    <div class="oss-component" id="component_{{ loop.index0 }}">
        {%for about_object in group.list %}
            {% if loop.first %}
                {% if about_object.license_url.value %}
                    {% for lic_url in about_object.license_url.value %}
                    <p>License URL: <a href="{{lic_url}}
                            ">{{lic_url }}</a> </p>
                    {% endfor %}
                {% endif %}
            {% endif %}
            <li>
            {{ about_object.name.value }}{% if about_object.version.value %}  - Version
            {{ about_object.version.value }}{% endif %}
            </li>
            {% if about_object.copyright.value %}<pre>{{about_object.copyright.value}}</pre>{% endif %}
            {% if loop.last %}
            <pre>
            {% for lic_key in about_object.license_file.value %}
                {{about_object.license_file.value[lic_key]}}
            {% endfor %}
            </pre>
            {% endif %}
        {% endfor %}
    </div>
    <hr>
    </div>
    {% endfor %}
{% endfor %}
<hr>
```

In summary, you can start with simple, cosmetic customizations to the default_html.template, and gradually introduce a more complex structure to the attrib output to meet varying business requirements.

Run attrib to Generate a Product Attribution Notice Package

When you have generated the AboutCode Toolkit files by gen, you can then run attrib to generate your product attribution notice package. The official attrib parameters are defined here:

- https://github.com/nexB/aboutcode-toolkit/blob/develop/REFERENCE.rst

Here is an example of a attrib command:

about attrib --template /Users/harrypotter/myAboutFiles/my_attribution_template_v1.html /Users/harrypotter/myAboutFiles/
/Users/harrypotter/myAboutFiles/myProject-attribution-document.html

Note that this example attrib command does the following:

- Activates the --template option to specify a custom output template.
- Specifies the path of the AboutCode Toolkit files needed to generate the output document.
- Specifies the full path (include file name) of the output document to be generated.

A successful execution of attrib will create a .html (or .json depends on the template) file that is ready to use to meet your attribution requirements.

Using inventory to Generate a Software Inventory

Generate a Software Inventory of Your Codebase from AboutCode Toolkit Files

One of the major features of the ABOUT File specification is that the .ABOUT files are very simple text files that can be created, viewed and edited using any standard text editor. Your software development and maintenance processes may require or encourage your software developers to maintain .ABOUT files and/or associated text files manually. For example, when a developer addresses a software licensing issue with a component, it is appropriate to adjust the associated AboutCode

Toolkit files manually.

If your organization adopts the practice of manually creating and maintaining AboutCode Toolkit files, you can easily re-create your software inventory from your codebase using inventory. The official inventory parameters are defined here:

- https://github.com/nexB/aboutcode-toolkit/blob/develop/REFERENCE.rst

A successful execution of inventory will create a complete software inventory in .csv format or .json format based on defined format.