CycloneDX Python Library

Release 7.3.2

Paul Horton, Jan Kowalleck, Steve Springett, Patrick Dwyer

Apr 29, 2024

CONTENTS:

OWASP CycloneDX is a full-stack Bill of Materials (BOM) standard that provides advanced supply chain capabilities for cyber risk reduction.

This Python package provides data models, validators and more, to help you create/render/read CycloneDX documents.

This package is not designed for standalone use. It is a software library.

As of version `3.0.0` of this library, the internal data model was adjusted to allow CycloneDX VEX documents to be produced as per official examples linking VEX to a separate CycloneDX document.

If you're looking for a CycloneDX tool to run to generate (SBOM) software bill-of-materials documents, why not checkout CycloneDX Python or Jake.

CHAPTER

ONE

INSTALLATION

Install from pypi.org as you would any other Python module using your preferred package manager:

```
pip install cyclonedx-python-lib
```

CycloneDX-python-lib is also available from conda-forge.

1.1 Extras

The following extras are available when installing this package:

json-validation
: Install the optional dependencies needed for JSON validation.

xml-validation
: Install the optional dependencies needed for XML validation.

validation
: Install the optional dependencies needed for all supported validations.

They can be used when installing in order to include additional dependencies, e.g.:

```
pip install 'cyclonedx-python-lib[validation]'
```

CHAPTER

TWO

ARCHITECTURE

This library broadly is separated into three key functional areas:

1. **Model**: Internal models used to unify data.

   Note: As of version 4.0.0 of this library we support deserialization from JSON and XML as well as serialization to JSON and XML.

2. **Output**: Choose and configure an output which allows you to define output format as well as the CycloneDX schema version

When wishing to generate a BOM, the process is as follows:

1. **Generate a Model by either:**
   1. Programmatically using this library
   2. By deserializing from an existing CycloneDX BOM document
2. Output the Model using an *cyclonedx.output* instance that reflects the schema version and format you require

2.1 Modelling

You can create a BOM Model from either manually using the methods available directly on the *cyclonedx.model.bom.Bom* class, or deserialize a JSON/XML via `cyclonedx.model.bom.Bom.from_json()`/`cyclonedx.model.bom.Bom.from_xml()`

Vulnerabilities are supported by the Model as of version 0.3.0.

Note: Known vulnerabilities associated with Components can be sourced from various data sources, but this library will not source them for you. Perhaps look at Jake if you're interested in this.

2.1.1 Example BOM created programmatically

Note: It is recommended that you have a good understanding of the CycloneDX Schema before attempting to create a BOM programmatically with this library.

For the most up-to-date in-depth examples, look at our Unit Tests.

2.1.2 Example BOM created from existing CycloneDX BOM

Note: Supported from version 4.0.0 of this library.

Deserializing from a CycloneDX JSON BOM

Each model class in this library that is serializable provides a magic `from_json()` method.

See the example below to read and deserialize a JSON CycloneDX document. Note that reading the file and loading as JSON is the programmers responsibility.

```
import json
from cyclonedx.model.bom import Bom

with open('/path/to/my/cyclonedx.json') as input_json:
    deserialized_bom = Bom.from_json(data=json.loads(input_json.read()))
```

Deserializing from a CycloneDX XML BOM

Each model class in this library that is serializable provides a magic `from_xml()` method.

See the example below to read and deserialize a XML CycloneDX document. Note that reading the file and loading as XML is the programmers responsibility. Be careful to avoid XML vulnerabilities as documented here. It is recommended that you use a library such as *defusedxml* instead of the native *xml.etree.ElementTree*.

```
from xml.etree import ElementTree
from cyclonedx.model.bom import Bom

with open('/path/to/my/cyclonedx.xml') as input_xml:
    deserialized_bom = cast(Bom, Bom.from_xml(data=ElementTree.fromstring(input_xml.
↪read())))
```

2.2 Schema Support

This library has partial support for the CycloneDX specification (we continue to grow support).

The following sub-sections aim to explain what support this library provides and any known gaps in support. We do this by calling out support for data as defined in the latest CycloneDX standard specification, regardless of whether it is supported in prior versions of the CycloneDX schema.

2.2.1 Root Level Schema Support

Data Path	Supported	Notes
bom[@version	Yes	
bom[@serialN	Yes	
bom.metadata	Yes	Not supported: `lifecycles`
bom.components	Yes	Not supported: `modified` (as it is deprecated), `modelCard`, `data`, `signature`.
bom.services	Yes	Not supported: `signature`.
bom.externalRefe	Yes	
bom.dependencies	Yes	Since `2.3.0`
bom.compositions	No	
bom.properties	Yes	Supported when outputting to Schema Version >= 1.5. See schema specification bug 130
bom.vulnerabilit	Yes	Note: Prior to CycloneDX 1.4, these were present under `bom.components` via a schema extension. Note: As of `cyclonedx-python-lib >3.0.0`, Vulnerability are modelled differently
bom.annotations	No	
bom.formulation	No	
bom.declarations	No	
bom.definitions	No	
bom.signature	No	

2.2.2 Internal Model Schema Support

Internal Model	Supported?	Notes
`ComponentEvidence`	Yes	Not currently supported: `callstack`, `identity`, `occurrences`.
`DisjunctiveLicense`	Yes	Not currently supported: `@bom-ref`, `licensing`, `properties`.
`LicenseExpression`	Yes	Not currently supported: `@bom-ref`
`OrganizationalContact`	Yes	Not currently supported: `@bom-ref`
`OrganizationalEntity`	Yes	Not currently supported: `@bom-ref`

2.3 Outputting

Once you have an instance of a *cyclonedx.model.bom.Bom* you can produce output in either **JSON** or **XML** against any of the supported CycloneDX schema versions.

We provide two helper methods:

- Output to string (for you to do with as you require)
- Output directly to a filename you provide

By default output will be in XML at latest supported schema version - see `cyclonedx.output.LATEST_SUPPORTED_SCHEMA_VERSION`.

2.3.1 Supported CycloneDX Schema Versions

This library supports the following schema versions:

- 1.0 (XML) - *(note, 1.1 schema version has no support for JSON)*
- 1.1 (XML) - *(note, 1.1 schema version has no support for JSON)*
- 1.2 (XML, JSON)
- 1.3 (XML, JSON)
- 1.4 (XML, JSON) - the latest supported schema version

2.3.2 Outputting to JSON

The below example relies on the latest schema version, but sets the output format to JSON. Output is returned as a `str`.

```
from cyclonedx.output import get_instance, BaseOutput, OutputFormat

outputter: BaseOutput = get_instance(bom=bom, output_format=OutputFormat.JSON)
bom_json: str = outputter.output_as_string()
```

2.3.3 Outputting to XML

The below example relies on the default output format being XML, but overrides the schema version to 1.2. Output is written to the supplied filename.

```
from cyclonedx.output import get_instance, BaseOutput, SchemaVersion

outputter: BaseOutput = get_instance(bom=bom, schema_version=SchemaVersion.V1_2)
outputter.output_to_file(filename='/tmp/sbom-v1.2.xml')
```

CHAPTER

THREE

EXAMPLES

3.1 Complex Serialize

```python
# This file is part of CycloneDX Python Lib
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
# SPDX-License-Identifier: Apache-2.0
# Copyright (c) OWASP Foundation. All Rights Reserved.

import sys
from typing import TYPE_CHECKING

from packageurl import PackageURL

from cyclonedx.exception import MissingOptionalDependencyException
from cyclonedx.factory.license import LicenseFactory
from cyclonedx.model import XsUri
from cyclonedx.model.bom import Bom
from cyclonedx.model.component import Component, ComponentType
from cyclonedx.model.contact import OrganizationalEntity
from cyclonedx.output import make_outputter
from cyclonedx.output.json import JsonV1Dot5
from cyclonedx.schema import OutputFormat, SchemaVersion
from cyclonedx.validation import make_schemabased_validator
from cyclonedx.validation.json import JsonStrictValidator

if TYPE_CHECKING:
    from cyclonedx.output.json import Json as JsonOutputter
    from cyclonedx.output.xml import Xml as XmlOutputter
```

(continues on next page)

(continued from previous page)

```
    from cyclonedx.validation.xml import XmlValidator


lc_factory = LicenseFactory()

# region build the BOM

bom = Bom()
bom.metadata.component = root_component = Component(
    name='myApp',
    type=ComponentType.APPLICATION,
    licenses=[lc_factory.make_from_string('MIT')],
    bom_ref='myApp',
)

component1 = Component(
    type=ComponentType.LIBRARY,
    name='some-component',
    group='acme',
    version='1.33.7-beta.1',
    licenses=[lc_factory.make_from_string('(c) 2021 Acme inc.')],
    supplier=OrganizationalEntity(
        name='Acme Inc',
        urls=[XsUri('https://www.acme.org')]
    ),
    bom_ref='myComponent@1.33.7-beta.1',
    purl=PackageURL('generic', 'acme', 'some-component', '1.33.7-beta.1')
)
bom.components.add(component1)
bom.register_dependency(root_component, [component1])

component2 = Component(
    type=ComponentType.LIBRARY,
    name='some-library',
    licenses=[lc_factory.make_from_string('GPL-3.0-only WITH Classpath-exception-2.0')]
)
bom.components.add(component2)
bom.register_dependency(component1, [component2])

# endregion build the BOM

# region JSON
"""demo with explicit instructions for SchemaVersion, outputter and validator"""

my_json_outputter: 'JsonOutputter' = JsonV1Dot5(bom)
serialized_json = my_json_outputter.output_as_string(indent=2)
print(serialized_json)
my_json_validator = JsonStrictValidator(SchemaVersion.V1_6)
try:
    validation_errors = my_json_validator.validate_str(serialized_json)
    if validation_errors:
        print('JSON invalid', 'ValidationError:', repr(validation_errors), sep='\n',␣
```

(continues on next page)

(continued from previous page)

```
↪file=sys.stderr)
        sys.exit(2)
    print('JSON valid')
except MissingOptionalDependencyException as error:
    print('JSON-validation was skipped due to', error)

# endregion JSON

print('', '=' * 30, '', sep='\n')

# region XML
"""demo with implicit instructions for SchemaVersion, outputter and validator.␣
↪TypeCheckers will catch errors."""

my_xml_outputter: 'XmlOutputter' = make_outputter(bom, OutputFormat.XML, SchemaVersion.
↪V1_6)
serialized_xml = my_xml_outputter.output_as_string(indent=2)
print(serialized_xml)
my_xml_validator: 'XmlValidator' = make_schemabased_validator(
    my_xml_outputter.output_format, my_xml_outputter.schema_version)
try:
    validation_errors = my_xml_validator.validate_str(serialized_xml)
    if validation_errors:
        print('XML invalid', 'ValidationError:', repr(validation_errors), sep='\n',␣
↪file=sys.stderr)
        sys.exit(2)
    print('XML valid')
except MissingOptionalDependencyException as error:
    print('XML-validation was skipped due to', error)

# endregion XML
```

3.2 Complex Deserialize

```
# This file is part of CycloneDX Python Lib
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
# SPDX-License-Identifier: Apache-2.0
# Copyright (c) OWASP Foundation. All Rights Reserved.
```

(continues on next page)

(continued from previous page)

```

import sys
from json import loads as json_loads
from typing import TYPE_CHECKING

from defusedxml import ElementTree as SafeElementTree  # type:ignore[import-untyped]

from cyclonedx.exception import MissingOptionalDependencyException
from cyclonedx.model.bom import Bom
from cyclonedx.schema import OutputFormat, SchemaVersion
from cyclonedx.validation import make_schemabased_validator
from cyclonedx.validation.json import JsonStrictValidator

if TYPE_CHECKING:
    from cyclonedx.validation.xml import XmlValidator

# region JSON

json_data = """{
  "$schema": "http://cyclonedx.org/schema/bom-1.6.schema.json",
  "bomFormat": "CycloneDX",
  "specVersion": "1.6",
  "serialNumber": "urn:uuid:88fabcfa-7529-4ba2-8256-29bec0c03900",
  "version": 1,
  "metadata": {
    "timestamp": "2024-02-10T21:38:53.313120+00:00",
    "tools": [
      {
        "vendor": "CycloneDX",
        "name": "cyclonedx-python-lib",
        "version": "6.4.1",
        "externalReferences": [
          {
            "type": "build-system",
            "url": "https://github.com/CycloneDX/cyclonedx-python-lib/actions"
          },
          {
            "type": "distribution",
            "url": "https://pypi.org/project/cyclonedx-python-lib/"
          },
          {
            "type": "documentation",
            "url": "https://cyclonedx-python-library.readthedocs.io/"
          },
          {
            "type": "issue-tracker",
            "url": "https://github.com/CycloneDX/cyclonedx-python-lib/issues"
          },
          {
            "type": "license",
            "url": "https://github.com/CycloneDX/cyclonedx-python-lib/blob/main/LICENSE"
          },
```

(continues on next page)

(continued from previous page)

```
          {
            "type": "release-notes",
            "url": "https://github.com/CycloneDX/cyclonedx-python-lib/blob/main/
↪CHANGELOG.md"
          },
          {
            "type": "vcs",
            "url": "https://github.com/CycloneDX/cyclonedx-python-lib"
          },
          {
            "type": "website",
            "url": "https://github.com/CycloneDX/cyclonedx-python-lib/#readme"
          }
        ]
      }
    ],
    "component": {
      "bom-ref": "myApp",
      "name": "myApp",
      "type": "application",
      "licenses": [
        {
          "license": {
            "id": "MIT"
          }
        }
      ]
    }
  },
  "components": [
    {
      "bom-ref": "myComponent@1.33.7-beta.1",
      "type": "library",
      "group": "acme",
      "name": "some-component",
      "version": "1.33.7-beta.1",
      "purl": "pkg:generic/acme/some-component@1.33.7-beta.1",
      "licenses": [
        {
          "license": {
            "name": "(c) 2021 Acme inc."
          }
        }
      ],
      "supplier": {
        "name": "Acme Inc",
        "url": [
          "https://www.acme.org"
        ]
      }
    },
    {
```

(continues on next page)

(continued from previous page)

```
      "bom-ref": "some-lib",
      "type": "library",
      "name": "some-library",
      "licenses": [
        {
          "expression": "GPL-3.0-only WITH Classpath-exception-2.0"
        }
      ]
    }
  ],
  "dependencies": [
    {
      "ref": "some-lib"
    },
    {
      "dependsOn": [
        "myComponent@1.33.7-beta.1"
      ],
      "ref": "myApp"
    },
    {
      "dependsOn": [
        "some-lib"
      ],
      "ref": "myComponent@1.33.7-beta.1"
    }
  ]
}"""
my_json_validator = JsonStrictValidator(SchemaVersion.V1_6)
try:
    validation_errors = my_json_validator.validate_str(json_data)
    if validation_errors:
        print('JSON invalid', 'ValidationError:', repr(validation_errors), sep='\n',␣
↪file=sys.stderr)
        sys.exit(2)
    print('JSON valid')
except MissingOptionalDependencyException as error:
    print('JSON-validation was skipped due to', error)
bom_from_json = Bom.from_json(  # type: ignore[attr-defined]
    json_loads(json_data))
print('bom_from_json', repr(bom_from_json))

# endregion JSON

print('', '=' * 30, '', sep='\n')

# endregion XML

xml_data = """<?xml version="1.0" ?>
<bom xmlns="http://cyclonedx.org/schema/bom/1.6"
  serialNumber="urn:uuid:88fabcfa-7529-4ba2-8256-29bec0c03900"
  version="1"
```

(continues on next page)

(continued from previous page)

```
>
  <metadata>
    <timestamp>2024-02-10T21:38:53.313120+00:00</timestamp>
    <tools>
      <tool>
        <vendor>CycloneDX</vendor>
        <name>cyclonedx-python-lib</name>
        <version>6.4.1</version>
        <externalReferences>
          <reference type="build-system">
            <url>https://github.com/CycloneDX/cyclonedx-python-lib/actions</url>
          </reference>
          <reference type="distribution">
            <url>https://pypi.org/project/cyclonedx-python-lib/</url>
          </reference>
          <reference type="documentation">
            <url>https://cyclonedx-python-library.readthedocs.io/</url>
          </reference>
          <reference type="issue-tracker">
            <url>https://github.com/CycloneDX/cyclonedx-python-lib/issues</url>
          </reference>
          <reference type="license">
            <url>https://github.com/CycloneDX/cyclonedx-python-lib/blob/main/LICENSE</
↪url>
          </reference>
          <reference type="release-notes">
            <url>https://github.com/CycloneDX/cyclonedx-python-lib/blob/main/CHANGELOG.md
↪</url>
          </reference>
          <reference type="vcs">
            <url>https://github.com/CycloneDX/cyclonedx-python-lib</url>
          </reference>
          <reference type="website">
            <url>https://github.com/CycloneDX/cyclonedx-python-lib/#readme</url>
          </reference>
        </externalReferences>
      </tool>
    </tools>
    <component type="application" bom-ref="myApp">
      <name>myApp</name>
      <licenses>
        <license>
          <id>MIT</id>
        </license>
      </licenses>
    </component>
  </metadata>
  <components>
    <component type="library" bom-ref="myComponent@1.33.7-beta.1">
      <supplier>
        <name>Acme Inc</name>
        <url>https://www.acme.org</url>
```

(continues on next page)

(continued from previous page)

```
      </supplier>
      <group>acme</group>
      <name>some-component</name>
      <version>1.33.7-beta.1</version>
      <licenses>
        <license>
          <name>(c) 2021 Acme inc.</name>
        </license>
      </licenses>
      <purl>pkg:generic/acme/some-component@1.33.7-beta.1</purl>
    </component>
    <component type="library" bom-ref="some-lib">
      <name>some-library</name>
      <licenses>
        <expression>GPL-3.0-only WITH Classpath-exception-2.0</expression>
      </licenses>
    </component>
  </components>
  <dependencies>
    <dependency ref="some-lib"/>
    <dependency ref="myApp">
      <dependency ref="myComponent@1.33.7-beta.1"/>
    </dependency>
    <dependency ref="myComponent@1.33.7-beta.1">
      <dependency ref="some-lib"/>
    </dependency>
  </dependencies>
</bom>"""
my_xml_validator: 'XmlValidator' = make_schemabased_validator(OutputFormat.XML,␣
↪SchemaVersion.V1_6)
try:
    validation_errors = my_xml_validator.validate_str(xml_data)
    if validation_errors:
        print('XML invalid', 'ValidationError:', repr(validation_errors), sep='\n',␣
↪file=sys.stderr)
        sys.exit(2)
    print('XML valid')
except MissingOptionalDependencyException as error:
    print('XML-validation was skipped due to', error)
bom_from_xml = Bom.from_xml(  # type: ignore[attr-defined]
    SafeElementTree.fromstring(xml_data))
print('bom_from_xml', repr(bom_from_xml))

# endregion XML

print('', '=' * 30, '', sep='\n')

print('assert bom_from_json equals bom_from_xml')
assert bom_from_json == bom_from_xml, 'expected to have equal BOMs from JSON and XML'
```

CHAPTER

FOUR

CONTRIBUTING

Pull requests are welcome. But please read the CycloneDX contributing guidelines first.

4.1 Setup

This project uses poetry. Have it installed and setup first.

To install dev-dependencies and tools:

```
poetry install --all-extras
```

4.2 Code style

This project uses PEP8 Style Guide for Python Code. This project loves sorted imports. Get it all applied via:

```
poetry run isort .
poetry run autopep8 -ir cyclonedx/ tests/ typings/ examples/
```

This project prefers `f'strings'` over `'string'.format()`. This project prefers `'single quotes'` over `"double quotes"`. This project prefers `lower_snake_case` variable names.

4.3 Documentation

This project uses Sphinx to generate documentation which is automatically published to readthedocs.io.

Source for documentation is stored in the `docs` folder in RST format.

You can generate the documentation locally by running:

```
cd docs
pip install -r requirements.txt
make html
```

4.4 Testing

Run all tests in dedicated environments, via:

```
poetry run tox run
```

4.5 Sign off your commits

Please sign off your commits, to show that you agree to publish your changes under the current terms and licenses of the project , and to indicate agreement with Developer Certificate of Origin (DCO).

```
git commit --signed-off ...
```

4.6 Pre-commit hooks

If you like to take advantage of pre-commit hooks, you can do so to cover most of the topics on this page when contributing.

```
pre-commit install
```

All our pre-commit checks will run locally before you can commit!

CHAPTER

FIVE

SUPPORT

If you run into issues utilising this library, please raise a GitHub Issue. When raising an issue please include as much detail as possible including:

- Version `cyclonedx-python-lib` you have installed
- Input(s)
- Expected Output(s)
- Actual Output(s)

5.1 Python Version Support

We endeavour to support all functionality for all current actively supported Python versions. However, some features may not be possible/present in older Python versions due to their lack of support - which are noted below.

CHAPTER

SIX

CHANGELOG

6.1 CHANGELOG

6.1.1 v7.3.2 (2024-04-26)

Fix

- fix: properly sort components based on all properties (#599)

reverts #587 - as this one introduced errors fixes #598 fixes #586

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> Signed-off-by: Paul Horton <paul.horton@owasp.org> Co-authored-by: Paul Horton <paul.horton@owasp.org> (``8df488c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8df488cb422a6363421fee39714df4e8e8e7a593>`_)

6.1.2 v7.3.1 (2024-04-22)

Chore

- chore: semantic-release git commit/sign valid email address

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``d437c40`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d437c40caa70071f0fcfe4e3c970370ee32d4aba>`_)

Fix

- fix: include all fields of `Component` in `__lt__` function for #586 (#587)

Fixes #586.

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``d784685`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d7846850d1ad33184d1d58b59fdf41a778d05900>`_)

6.1.3 v7.3.0 (2024-04-19)

Feature

- feat: license factory set `acknowledgement` (#593)

add a parameter to `LicenseFactory.make_*()` methods, to set the `LicenseAcknowledgement`.

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*7ca2455*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7ca2455018d0e191afaaa2fd136a7e4d5b325ec6>`_)

6.1.4 v7.2.0 (2024-04-19)

Feature

- feat: disjunctive license acknowledgement (#591)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*9bf1839*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9bf1839859a244e790e91c3e1edd82d333598d60>`_)

Unknown

- tests: add meaningful names to validation tests (#588)

When packaging cyclonedx-python-lib for a Linux distribution, it's pretty common that some JSON validation tests fail. 1

Due to the large number of combinations and the fact that these tests are consecutively numbered, it has been tedious to figure out which tests are exactly failing and why. This in turn makes it difficult to decide which tests to disable or report upstream.

Append meaningful names to validation tests so that instead of e.g.:

```
[...]::TestJsonValidator::test_validate_no_none_001
[...]::TestJsonValidator::test_validate_no_none_002
[...]::TestJsonValidator::test_validate_no_none_003
[...]::TestJsonValidator::test_validate_no_none_004
[...]::TestJsonValidator::test_validate_no_none_005
[...]::TestJsonValidator::test_validate_no_none_006
[...]::TestJsonValidator::test_validate_no_none_007
[...]::TestJsonValidator::test_validate_no_none_008
```

the tests are named:

```
[...]::TestJsonValidator::test_validate_no_none_001_valid_component_swid_1_6
[...]::TestJsonValidator::test_validate_no_none_002_valid_machine_learning_
↪considerations_env_1_6
[...]::TestJsonValidator::test_validate_no_none_003_valid_metadata_tool_1_6
[...]::TestJsonValidator::test_validate_no_none_004_valid_patch_1_6
[...]::TestJsonValidator::test_validate_no_none_005_valid_empty_components_1_6
[...]::TestJsonValidator::test_validate_no_none_006_valid_properties_1_6
[...]::TestJsonValidator::test_validate_no_none_007_valid_service_1_6
[...]::TestJsonValidator::test_validate_no_none_008_valid_metadata_author_1_6
```

Signed-off-by: Claudia <claui@users.noreply.github.com> (``ae3f79c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ae3f79cbaeecda94948bff6a64ab797c5ddd934a>`_)

- doc: poor merge resolved

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``a498faa`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a498faaab248d0512bad9e66afbd8fb1d6c42a66>`_)

6.1.5 v7.1.0 (2024-04-10)

Documentation

- docs: missing schema support table & update schema support to reflect version 7.0.0 (#584)

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``d230e67`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d230e67188661a5fb94730e52bf59c11c965c8d7>`_)

Feature

- feat: support `bom.properties` for CycloneDX v1.5+ (#585)

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``1d1c45a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1d1c45ac82c7927acc388489228a9b5990f68aa7>`_)

6.1.6 v7.0.0 (2024-04-09)

Breaking

- feat!: Support for CycloneDX v1.6
- added draft v1.6 schemas and boilerplate for v1.6

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- re-generated test snapshots for v1.6

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- note `bom.metadata.manufacture` as deprecated

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- work on `bom.metadata` for v1.6

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- Deprecated `.component.author`. Added `.component.authors` and `.component.manufacturer`

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- work to add `.component.omniborid` - but tests deserialisation tests fail due to schema differences (`.component.author` not in 1.6)

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- work to get deserialization tests passing

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- chore(deps): bump `py-serializable` to >=1.0.3 to resolve issues with deserialization to XML

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- imports tidied

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- properly added `.component.swhid`

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- add `.component.cryptoProperties` - with test failures for SchemaVersion < 1.6

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- typing and bandit ignores

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- coding standards

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- test filtering

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- coding standards

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- additional tests to increase code coverage

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- corrected CryptoMode enum

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- coding standards

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- Added `address` to `organizationalEntity`

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- Added `address` to `organizationalEntity`

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- raise `UserWarning` in `.component.version` has length > 1024

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- coding standards and typing

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- add `acknowledgement` to `LicenseExpression` (#582)

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- more proper way to filter test cases

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- update schema to published versions

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- fetch schema 1.6 JSON

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- fetch test data for CDX 1.6

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- reformat

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- reformat

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- refactor

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- style

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- refactor

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- docs

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

Signed-off-by: Paul Horton <paul.horton@owasp.org> Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> Co-authored-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*8bbdf46*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8bbdf461434ab66673a496a8305c2878bf5c88da>`_)

Chore

- chore(deps-dev): update autopep8 requirement from 2.0.4 to 2.1.0 (#573)

Updates the requirements on autopep8 to permit the latest version.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: autopep8 dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*35749c6*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/35749c6cd18ebb8911b7cefac8a381d2ee57177a>`_)

- chore(deps-dev): update tox requirement from 4.14.1 to 4.14.2 (#574)

Updates the requirements on tox to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: tox dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*d60f457*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d60f4570621246ce3d68e7f2e7f1aa831fb818f0>`_)

6.1.7 v6.4.4 (2024-03-18)

Chore

- chore(deps-dev): update coverage requirement from 7.4.3 to 7.4.4 (#570)

Updates the requirements on coverage to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: coverage dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*3a2e427*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3a2e427ba9967f11c15cd1a47c59a933b699c87b>`_)

- chore(deps): bump python-semantic-release/python-semantic-release (#564)

Bumps python-semantic-release/python-semantic-release from 8.5.1 to 9.1.1.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: python-semantic-release/python-semantic-release dependency-type: direct:production update-type: version-update:semver-major …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*d20a590*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d20a5902582facab0636e9ff8a261edcaf886a3e>`_)

- chore(deps-dev): update tox requirement from 4.13.0 to 4.14.1 (#567)

Updates the requirements on tox to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: tox dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*2dcc60e*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2dcc60e53ec66d642c728596ff25fed4df5659a0>`_)

- chore(deps-dev): update bandit requirement from 1.7.7 to 1.7.8 (#566)

Updates the requirements on bandit to permit the latest version.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: bandit dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*eb1a252*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/eb1a2525c09e0dd10f11ff83b451a4db4fb00d9b>`_)

- chore(deps-dev): update mypy requirement from 1.8.0 to 1.9.0 (#565)

Updates the requirements on mypy to permit the latest version.

- Changelog
- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*3ce0f3a*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3ce0f3a373d9f1b07af50d9b707f766ea446e518>`_)

Fix

- fix: wrong extra name for xml validation (#571)

Signed-off-by: Christoph Reiter <reiter.christoph@gmail.com> (``*10e38e2*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/10e38e25095de4b2dafbfcd1fd81dce7a9c0f124>`_)

6.1.8 v6.4.3 (2024-03-04)

Chore

- chore(deps-dev): update ddt requirement from 1.7.1 to 1.7.2 (#563)

Updates the requirements on ddt to permit the latest version.

- Release notes

- Commits

---

updated-dependencies:

- dependency-name: ddt dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``53cb8a9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/53cb8a9aa2630e992467525ff246a0f6e6759100>`_)

Fix

- fix: serialization of `model.component.Diff` (#557)

Fixes #556

Signed-off-by: rcross-lc <151086351+rcross-lc@users.noreply.github.com> Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> Co-authored-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``22fa873`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/22fa8734bf1a3a8789ad7578bfa0c86cf0a49d4a>`_)

6.1.9 v6.4.2 (2024-03-01)

Build

- build: use poetry v1.8.1 (#560)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``6f81dfa`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6f81dfaed32b76f251647f6291791e714ab158a3>`_)

Chore

- chore(deps-dev): update coverage requirement from 7.4.1 to 7.4.3 (#558)

Updates the requirements on coverage to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: coverage dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``2b7f261`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2b7f261585faa6237e635b18d5ecaf03d81439ba>`_)

- chore(deps): bump Gr1N/setup-poetry from 8 to 9 (#555)

Bumps Gr1N/setup-poetry from 8 to 9.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: Gr1N/setup-poetry dependency-type: direct:production update-type: version-update:semver-major ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``178ce32`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/178ce32c0da822b8f1b4d13b427d6f21ea252b59>`_)

- chore(deps-dev): update tox requirement from 4.12.1 to 4.13.0 (#553)

Updates the requirements on tox to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: tox dependency-type: direct:development ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``77fb2ec`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/77fb2ec3593fac577a48894f329a77a7ac6d417c>`_)

- chore(deps-dev): update flake8-quotes requirement from 3.3.2 to 3.4.0 (#552)

Updates the requirements on flake8-quotes to permit the latest version.

- Commits

---

updated-dependencies:

- dependency-name: flake8-quotes dependency-type: direct:development ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``cd8e67c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/cd8e67c15ae09a07f51f15855c1ae05734352f52>`_)

- chore(deps-dev): update flake8-bugbear requirement (#549)

Updates the requirements on flake8-bugbear to permit the latest version.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: flake8-bugbear dependency-type: direct:development ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``153d83e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/153d83e5a18a2696d49884319fd156628a19cd7b>`_)

Documentation

- docs: update architecture description and examples (#550)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``a19fd28`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a19fd2828355ae031164ef7a0dda2a8ea2365108>`_)

- docs: exclude internal docs from rendering (#545)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``7e55dfe`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7e55dfe213cb2a88b3686f9e8bf93cf4642a2ccd>`_)

Unknown

- docs

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``63cff7e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/63cff7ee697c9d5fb96da3c8c16f7c9bc7b34e58>`_)

- docs (#546)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``b0e5b43`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b0e5b43880e17ec6ce23d5d4e1e7a9a2547c1e79>`_)

6.1.10 v6.4.1 (2024-01-30)

Chore

- chore(deps-dev): update bandit requirement from 1.7.6 to 1.7.7 (#542)

Updates the requirements on bandit to permit the latest version.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: bandit dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``0d159c2`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0d159c29cab7cd57e2028a302ef24f1947de235d>`_)

- chore(deps-dev): update coverage requirement from 7.4.0 to 7.4.1 (#541)

Updates the requirements on coverage to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: coverage dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``fa82a24`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fa82a2413f1aa350d16ad3ac0c5163da97e29e34>`_)

Documentation

- docs: ship docs with `sdist` build (#544)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``52ef01c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/52ef01c99319d5aed950e7f6ef6fcfe731ac8b2f>`_)

- docs: refactor example

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``c1776b7`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c1776b718b81cf72ef0c0251504e0d3631e30b17>`_)

Fix

- fix: `model.BomRef` no longer equal to unset peers (#543)

  fixes #539

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``1fd7fee`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1fd7fee9dec888c10087921f2e5a7a60062fb419>`_)

Unknown

- tests: fetched schema 1.5 test data from spec (#536)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``394cc87`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/394cc87b3247b6f57af4073f5548f1c5eada2b9b>`_)

6.1.11 v6.4.0 (2024-01-22)

Chore

- chore(deps-dev): update tox requirement from 4.12.0 to 4.12.1 (#533)

Updates the requirements on tox to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: tox dependency-type: direct:development ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``74094d7`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/74094d70c15afdd9991f8b731d318f66b686cf62>`_)

- chore(deps-dev): update flake8-bugbear requirement (#534)

Updates the requirements on flake8-bugbear to permit the latest version.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: flake8-bugbear dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``6e6f374`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6e6f374ba282a67c9458b414704a3d86f4b593b4>`_)

- chore: doc flake8 config

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``bd4c078`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bd4c0781139bc93e28438390650ef1c7484597bb>`_)

- chore(deps-dev): update tox requirement from 4.11.4 to 4.12.0 (#530)

Updates the requirements on tox to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: tox dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``130918a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/130918a78d003255f1d80e6fe2031752c3baa6d1>`_)

Documentation

- docs: add OpenSSF Best Practices shield (#532)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``59c4381`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/59c43814b07db0aa881d87192939eb93e79b0cc2>`_)

Feature

- feat: support `py-serializable` v1.0 (#531)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``e1e7277`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e1e72777d8a355c6854f4d9eb26c1e2083c806df>`_)

6.1.12 v6.3.0 (2024-01-06)

Chore

- chore(deps-dev): update flake8 requirement from 6.1.0 to 7.0.0 (#528)

Updates the requirements on flake8 to permit the latest version.

- Commits

---

updated-dependencies:

- dependency-name: flake8 dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``6b7ed78`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6b7ed786845d21bf079c0a636d9e689ce278644c>`_)

- chore(deps-dev): update ddt requirement from 1.7.0 to 1.7.1 (#527)

Updates the requirements on ddt to permit the latest version.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: ddt dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``9a58e7e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9a58e7ee921a077425ee45f23e9cfbb8341d7ef5>`_)

Documentation

- docs: add `Documentation` url to project meta

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``1080b73`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1080b7387a0bbc49a067cd2efefb1545470947e5>`_)

- docs: add `Documentation` url to project meta

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``c4288b3`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c4288b35e0e1050f0982f7492cfcd3bea34b445c>`_)

Feature

- feat: enable dependency `py-serializable 0.17` (#529)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``9f24220`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9f24220029cd18cd191f63876899cd86be52dce1>`_)

6.1.13 v6.2.0 (2023-12-31)

Build

- build: allow additional major-version RC branch patterns

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``f8af156`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f8af156c9c38f737b7067722d2a96f8a2a4fcb48>`_)

Chore

- chore(deps-dev): update coverage requirement from 7.3.3 to 7.4.0 (#524)

Updates the requirements on coverage to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: coverage dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``9bcc223`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9bcc223b783306cf2255b3910acf5518d7ea223c>`_)

- chore(deps-dev): update mypy requirement from 1.7.1 to 1.8.0 (#521)

Updates the requirements on mypy to permit the latest version.

- Changelog
- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``720046e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/720046e2f69c64216b5ef847ad5f76a95f450a8f>`_)

Documentation

- docs: fix typo

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``2563996`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/25639967c93ad464e486f2fe6a148b3be439f43d>`_)

- docs: update intro and description

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``f0bd05d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f0bd05dc854b5b71421b82cfb527fcb8f41a7c4a>`_)

- docs: buld docs on ubuntu22.04 python311

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``b3e9ab7`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b3e9ab77696f2ee763f1746f8142bdf471477c39>`_)

Feature

- feat: allow `lxml` requirement in range of `>=4,<6` (#523)

Updates the requirements on lxml to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: lxml dependency-type: direct:production …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``7d12b9a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7d12b9a9f7a2fdc5e6bb12f891c6f4291e20e65e>`_)

Unknown

- docs

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``7dcd166`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7dcd16621002713dcf1ce8e17bc5762320fae4fa>`_)

6.1.14 v6.1.0 (2023-12-22)

Chore

- chore: update maintainers

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``87c72d7`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/87c72d7f589faea67c5f90f041531468f8ae480c>`_)

- chore(deps): bump python-semantic-release/python-semantic-release (#515)

Bumps python-semantic-release/python-semantic-release from 8.5.0 to 8.5.1.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: python-semantic-release/python-semantic-release dependency-type: direct:production update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``0f56ec4`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0f56ec471268d0b27c5956b93021a982945873a2>`_)

- chore(deps-dev): update coverage requirement from 7.3.2 to 7.3.3 (#517)

Updates the requirements on coverage to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: coverage dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``a57e2f6`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a57e2f6ee14d015e58e2175dcbb087d971731f92>`_)

- chore(deps-dev): update isort requirement from 5.13.0 to 5.13.2 (#516)

Updates the requirements on isort to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: isort dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``84874a3`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/84874a39610b0108335413da23f50b2911c20c78>`_)

Feature

- feat: add function to map python `hashlib` algorithms to CycloneDX (#519)

new API: `model.HashType.from_hashlib_alg()`

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``81f8cf5`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/81f8cf59b1f40ffbd213789a8b1b621a01e3f631>`_)

6.1.15 v6.0.0 (2023-12-10)

Breaking

- feat!: v6.0.0 (#492)

Breaking Changes

- Removed symbols that were already marked as deprecated (via #493)
- Removed symbols in `parser.*` (#489 via #495)
- Removed `output.LATEST_SUPPORTED_SCHEMA_VERSION` (#491 via #494)
- Serialization of unsupported enum values might downgrade/migrate/omit them (#490 via #496) Handling might raise warnings if a data loss occurred due to omitting. The result is a guaranteed valid XML/JSON, since no (enum-)invalid values are rendered.

- Serialization of any `model.component.Component` with unsupported `type` raises `exception.serialization.SerializationOfUnsupportedComponentTypeException` (#490 via #496)
- Object `model.bom_ref.BomRef`'s property `value` defaults to `Null`, was arbitrary `UUID` (#504 via #505) This change does not affect serialization. All `bom-ref`s are guaranteed to have unique values on rendering.
- Removed helpers from public API (#503 via #506)

Added

- Basic support for CycloneDX 1.5 (#404 via #488)
  - No data models were enhanced nor added, yet. Pull requests to add functionality are welcome.
  - Existing enumerable got new cases, to reflect features of CycloneDX 1.5 (#404 via #488)
  - Outputters were enabled to render CycloneDX 1.5 (#404 via #488)

Tests

- Created (regression/unit/integration/functional) tests for CycloneDX 1.5 (#404 via #488)
- Created (regression/functional) tests for Enums' handling and completeness (#490 via #496)

Misc

- Bumped dependency `py-serializable@^0.16`, was `@^0.15` (via #496)

API Changes — the details for migration

- Added new sub-package `exception.serialization` (via #496)
- Removed class `models.ComparableTuple` (#503 via #506)
- Enum `model.ExternalReferenceType` got new cases, to reflect features for CycloneDX 1.5 (#404 via #488)
- Removed function `models.get_now_utc` (#503 via #506)
- Removed function `models.sha1sum` (#503 via #506)
- Enum `model.component.ComponentType` got new cases, to reflect features for CycloneDX 1.5 (#404 via #488)
- Removed `model.component.Component.__init__()`'s deprecated optional kwarg `namespace` (via #493) Use kwarg `group` instead.
- Removed `model.component.Component.__init__()`'s deprecated optional kwarg `license_str` (via #493) Use kwarg `licenses` instead.
- Removed deprecated method `model.component.Component.get_namespace()` (via #493)
- Removed class `models.dependency.DependencyDependencies` (#503 via #506)
- Removed `model.vulnerability.Vulnerability.__init__()`'s deprecated optional kwarg `source_name` (via #493) Use kwarg `source` instead.
- Removed `model.vulnerability.Vulnerability.__init__()`'s deprecated optional kwarg `source_url` (via #493) Use kwarg `source` instead.

- Removed `model.vulnerability.Vulnerability.__init__()`'s deprecated optional kwarg `recommendations` (via #493) Use kwarg `recommendation` instead.
- Removed `model.vulnerability.VulnerabilityRating.__init__()`'s deprecated optional kwarg `score_base` (via #493) Use kwarg `score` instead.
- Enum `model.vulnerability.VulnerabilityScoreSource` got new cases, to reflect features for CycloneDX 1.5 (#404 via #488)
- Removed `output.LATEST_SUPPORTED_SCHEMA_VERSION` (#491 via #494)
- Removed deprecated function `output.get_instance()` (via #493) Use function `output.make_outputter()` instead.
- Added new class `output.json.JsonV1Dot5`, to reflect CycloneDX 1.5 (#404 via #488)
- Added new item to dict `output.json.BY_SCHEMA_VERSION`, to reflect CycloneDX 1.5 (#404 via #488)
- Added new class `output.xml.XmlV1Dot5`, to reflect CycloneDX 1.5 (#404 via #488)
- Added new item to dict `output.xml.BY_SCHEMA_VERSION`, to reflect CycloneDX 1.5 (#404 via #488)
- Removed class `parser.ParserWarning` (#489 via #495)
- Removed class `parser.BaseParser` (#489 via #495)
- Enum `schema.SchemaVersion` got new case `V1_5`, to reflect CycloneDX 1.5 (#404 via #488)

Signed-off-by: Johannes Feichtner <johannes@web-wack.at> Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> Signed-off-by: semantic-release <semantic-release> Co-authored-by: Johannes Feichtner <343448+Churro@users.noreply.github.com> Co-authored-by: semantic-release <semantic-release> (``*74865f8*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/74865f8e498c9723c2ce3556ceecb6a3cfc4c490>`_)

Chore

- chore(deps): bump python-semantic-release/python-semantic-release (#509)

Bumps python-semantic-release/python-semantic-release from 8.0.8 to 8.5.0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: python-semantic-release/python-semantic-release dependency-type: direct:production update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*9ed9ab1*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9ed9ab129e5123b061a9cd358d418f026d2e8b7a>`_)

- chore(deps-dev): update isort requirement from 5.12.0 to 5.13.0 (#512)

Updates the requirements on isort to permit the latest version.

- Release notes
- Changelog

- Commits

---

updated-dependencies:

- dependency-name: isort dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*0eba631*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0eba631c628faab454eabba66755d311606c536a>`_)

- chore(deps-dev): update bandit requirement from 1.7.5 to 1.7.6 (#510)

Updates the requirements on bandit to permit the latest version.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: bandit dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*153b07a*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/153b07a28047c408e0dc442330aa1505e74c175e>`_)

- chore(deps): bump actions/setup-python from 4 to 5 (#508)

Bumps actions/setup-python from 4 to 5.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: actions/setup-python dependency-type: direct:production update-type: version-update:semver-major …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*4e3e0e0*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/4e3e0e0e873ff45a4d6605728cb1160cd912e3cf>`_)

- chore(deps): update sphinx-rtd-theme requirement (#499)

Updates the requirements on sphinx-rtd-theme to permit the latest version.

- Changelog
- Commits

---

updated-dependencies:

- dependency-name: sphinx-rtd-theme dependency-type: direct:production …

Signed-off-by: dependabot[bot] <support@github.com> Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*5d6dd41*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5d6dd417e0c13e596dab6c76b610788bbbb96093>`_)

- chore(deps-dev): update flake8-bugbear requirement (#500)

Updates the requirements on flake8-bugbear to permit the latest version.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: flake8-bugbear dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``e9a12b9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e9a12b93a9866dcb6d9a48396a7c5378d5c5f5e5>`_)

- chore(deps): update py-serializable requirement (#501)

Updates the requirements on py-serializable to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: py-serializable dependency-type: direct:production …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``04435ab`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/04435abe926b6fa877bd0194733ae87b3bad9610>`_)

- chore(deps-dev): update tox requirement from 4.11.3 to 4.11.4 (#502)

Updates the requirements on tox to permit the latest version.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: tox dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``8bf0e39`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8bf0e39f62c57c8afb6b7c152156e7af1f02bd5d>`_)

6.1.16 v5.2.0 (2023-12-02)

Chore

- chore(deps-dev): update mypy requirement from 1.7.0 to 1.7.1 (#487)

Updates the requirements on mypy to permit the latest version.

- Changelog
- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``78957e6`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/78957e6592be2106de7933f5b54a0916194980e3>`_)

- chore(deps-dev): update mypy requirement from 1.6.1 to 1.7.0 (#484)

Updates the requirements on mypy to permit the latest version.

- Changelog
- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``c716ba3`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c716ba3751017e2f88367b660dbc11866c2feb1d>`_)

- chore(deps-dev): update ddt requirement from 1.6.0 to 1.7.0 (#483)

Updates the requirements on ddt to permit the latest version.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: ddt dependency-type: direct:development …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``8a1f7b9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8a1f7b9678e888654a373965b7902428525f7d60>`_)

- chore: mograte dev-dependencies to new poetry layout (#482)

see https://python-poetry.org/docs/managing-dependencies/#dependency-groups

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``a85585c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a85585cf5e445ba5e67a027b4d1161911df6467d>`_)

- chore(deps-dev): update flake8-isort requirement from 6.1.0 to 6.1.1 (#481)

Updates the requirements on flake8-isort to permit the latest version.

- Changelog
- Commits

---

updated-dependencies:

- dependency-name: flake8-isort dependency-type: direct:development ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (`*fc74ddd*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fc74dddc8add79be31d3747ddce9241bce2e4fed>`_)

Documentation

- docs: keywaords & funding (#486)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (`*3189e59*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3189e59ff8e3d3d10f7b949b5a08397ff3d3642b>`_)

Feature

- feat: `model.XsUri` migrate control characters according to spec (#498)

fixes https://github.com/CycloneDX/cyclonedx-python-lib/issues/497

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (`*e490429*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e49042976f8577af4061c34394db270612488cdf>`_)

6.1.17 v5.1.1 (2023-11-02)

Fix

- fix: update own `externalReferences` (#480)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (`*edb3dde*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/edb3dde889c06755dd1963ed21dd803db3ea0dcc>`_)

6.1.18 v5.1.0 (2023-10-31)

Documentation

- docs: advance license docs

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (`*f61a730*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f61a7303de1d5dacf0917a1d66f5ebe0732ccd75>`_)

Feature

- feat: guarantee unique `BomRef`s in serialization result (#479)

Incorporate `output.BomRefDiscriminator` on serialization

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``a648775`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a648775bb5195621e17fdbae92950ab6d56a665a>`_)

6.1.19 v5.0.1 (2023-10-24)

Chore

- chore(deps): bump python-semantic-release/python-semantic-release (#474)

Bumps python-semantic-release/python-semantic-release from 8.0.8 to 8.3.0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: python-semantic-release/python-semantic-release dependency-type: direct:production update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``9c3ffac`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9c3ffac34e89610ccc4f9701444127e1e6f5ee07>`_)

- chore: make `pyproject` parsable by dependabot (#477)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``c4eaaa5`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c4eaaa54d98da43d0cdbb19b5f61e06a21f1cc58>`_)

Documentation

- docs: revisit project meta (#475)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``c3254d0`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c3254d055f3cda96d2849222a0bba7be8cf486a3>`_)

- docs: fix RTFD build (#476)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``b9fcfb4`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b9fcfb40af366fdee7258ccb720e0fad27994824>`_)

Unknown

- "chore(deps): revert bump python-semantic-release/python-semantic-release (#474)"

This reverts commit 9c3ffac34e89610ccc4f9701444127e1e6f5ee07.

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*aae7304*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/aae73048c7aebe5920ec888225bdbde08111601b>`_)

6.1.20 v5.0.0 (2023-10-24)

Breaking

- feat!: v5.0.0 (#440)

6.1.21 BREAKING CHANGES

- Dropped support for python<3.8 (#436 via #441; enable #433)
- Reworked license related models, collections, and factories (#365 via #466)
- Behavior
  - Method `model.bom.Bom.validate()` will throw `exception.LicenseExpressionAlongWithOthersException`, if detecting invalid license constellation (#453 via #452)
  - Fixed tuple comparison when unequal lengths (via #461)
- API
  - Enum `schema.SchemaVersion` is no longer string-like (#442 via #447)
  - Enum `schema.OutputVersion` is no longer string-like (#442 via #447)
  - Abstract class `output.BaseOutput` requires implementation of new method `output_format` (#446 via #447)
  - Abstract method `output.BaseOutput.output_as_string()` got new optional parameter `indent` (#437 via #458)
  - Abstract method `output.BaseOutput.output_as_string()` accepts arbitrary kwargs (via #458, #462)
  - Removed class `factory.license.LicenseChoiceFactory` (via #466) The old functionality was integrated into `factory.license.LicenseFactory`.
  - Method `factory.license.LicenseFactory.make_from_string()`'s parameter `name_or_spdx` was renamed to `value` (via #466)
  - Method `factory.license.LicenseFactory.make_from_string()`'s return value can also be a `LicenseExpression` (#365 via #466) The behavior imitates the old `factory.license.LicenseChoiceFactory.make_from_string()`
  - Renamed class `module.License` to `module.license.DisjunctliveLicense` (#365 via #466)
  - Removed class `module.LicenseChoice` (#365 via #466) Use dedicated classes `module.license.DisjunctliveLicense` and `module.license.LicenseExpression` instead
  - All occurrences of `models.LicenseChoice` were replaced by `models.licenses.License` (#365 via #466)

  - All occurrences of `SortedSet[LicenseChoice]` were specialized to `models.license.LicenseRepository` (#365 via #466)

6.1.22 Fixed

- Serialization of multy-licenses (#365 via #466)
- Detect unused "dependent" components in `model.bom.validate()` (via #464)

6.1.23 Changed

- Updated latest supported list of supported SPDX license identifiers (via #433)
- Shipped schema files are moved to a protected space (via #433) These files were never intended for public use.
- XML output uses a default namespace, which makes results smaller. (#438 via #458)

6.1.24 Added

- Support for Python 3.12 (via #460)
- JSON- & XML-Validators (#432, #446 via #433, #448) The functionality might require additional dependencies, that can be installed with the extra "validation". See the docs in section "Installation" for details.
- JSON & XML can be generated in a more human-friendly form (#437, #438 via #458)
- Type hints, typings & overloads for better integration downstream (via #463)
- API
  - New function `output.make_outputter()` (via #469) This replaces the deprecated function `output.get_instance()`.
  - New sub-package `validation` (#432, #446 via #433, #448, #469, #468, #469)
  - New class `exception.MissingOptionalDependencyException` (#432 via #433)
  - New class `exception.LicenseExpressionAlongWithOthersException` (#453 via #452)
  - New dictionaries `output.{json,xml}.BY_SCHEMA_VERSION` (#446 via #447)
  - Existing implementations of class `output.BaseOutput` now have a new method `output_format` (#446 via #447)
  - Existing implementations of method `output.BaseOutput.output_as_string()` got new optional parameter `indent` (#437 via #458)
  - Existing implementations of method `output.BaseOutput.output_to_file()` got new optional parameter `indent` (#437 via #458)
  - New method `factory.license.LicenseFactory.make_with_expression()` (via #466)
  - New class `model.license.DisjunctiveLicense` (#365 via #466)
  - New class `model.license.LicenseExpression` (#365 via #466)
  - New class `model.license.LicenseRepository` (#365 via #466)
  - New class `serialization.LicenseRepositoryHelper` (#365 via #466)

6.1.25 Deprecated

- Function `output.get_instance()` might be removed, use `output.make_outputter()` instead (via #469)

6.1.26 Tests

- Added validation tests with official CycloneDX schema test data (#432 via #433)
- Use proper snapshots, instead of pseudo comparison (#437 via #464)
- Added regression test for bug #365 (via #466, #467)

6.1.27 Misc

- Dependencies: bumped `py-serializable@^0.15.0`, was `@^0.11.1` (via #458, #463, #464, #466)
- Style: streamlined quotes and strings (via #472)
- Chore: bumped internal dev- and QA-tools (#436 via #441, #472)
- Chore: added more QA tools to prevent common security issues (via #473)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> Signed-off-by: Jan Kowalleck <jan.kowalleck@owasp.org> Signed-off-by: semantic-release <semantic-release> Co-authored-by: semantic-release <semantic-release> (``*26b151c*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/26b151cba7d7d484f23ee7888444f09ad6d016b1>`_)

6.1.28 v4.2.3 (2023-10-16)

Chore

- chore: Update CONTRIBUTING.md

Signed-off-by: Jan Kowalleck <jan.kowalleck@owasp.org> (``*0ebaa21*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0ebaa216d43a1389362dbdb33f9b49f43a21ab66>`_)

Ci

- ci: publish coverage report to codacy (#439)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*0012a82*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0012a82382f9f33831a80aa0e26c0cbb7fd8984b>`_)

Fix

- fix: SPDX-expression-validation internal crashes are cought and handled (#471)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``5fa66a0`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5fa66a043818eb5747dbd630496c6d31f818c0ab>`_)

6.1.29 v4.2.2 (2023-09-14)

Chore

- chore: dont lock poetry (#431)

fixes #430

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``49b144b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/49b144be519705b03adc510ddcc6b9e4504b7a40>`_)

- chore(deps): bump actions/checkout from 3 to 4 (#429)

Bumps actions/checkout from 3 to 4.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: actions/checkout dependency-type: direct:production update-type: version-update:semver-major …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``a70754d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a70754d602e109538c06e06e59f563953c21ab1b>`_)

Documentation

- docs: fix shield in README

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``6a941b1`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6a941b1ef5cc0f9e956173cce7e9da57e8c6bf22>`_)

- docs(example): showcase `LicenseChoiceFactory` (#428)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``c56ec83`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c56ec8395dd203ac41fa6f4c43970a50c0e80efb>`_)

Fix

- fix: ship meta files (#434)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*3a1a8a5*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3a1a8a5c1cbe8d8989b4cb335269a02b5c6d4f38>`_)

6.1.30 v4.2.1 (2023-09-06)

Fix

- fix: `LicenseChoiceFactory.make_from_string()` prioritize SPDX id over expression (#427)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*e1bdfdd*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e1bdfddcfab97359fbde9f53dc65f56fc8ec4ba9>`_)

6.1.31 v4.2.0 (2023-09-06)

Chore

- chore(deps): bump python-semantic-release/python-semantic-release (#423)

Bumps python-semantic-release/python-semantic-release from 8.0.7 to 8.0.8.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: python-semantic-release/python-semantic-release dependency-type: direct:production update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*13e441d*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/13e441d581e2c419b46719148078155d44786e52>`_)

Feature

- feat: complete SPDX license expression (#425)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*e06f9fd*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e06f9fd2c30e8976766f326ff216103d2560cb9a>`_)

6.1.32 v4.1.0 (2023-08-27)

Chore

- chore: migrate to python-semantic-release8 (#421)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``14c501c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/14c501c1133c747e1a7dad6df8cad01a03f71a20>`_)

- chore: migrate to python-semantic-release8 (#420)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``0e35d88`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0e35d88b329bebe05f19748a23a31abf6295c933>`_)

- chore: migrate to python-semantic-release8 (#419)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``adf5a36`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/adf5a3668c7c9aa3e0478fd1eabf3b3163fae691>`_)

- chore(deps-dev): bump distlib from 0.3.6 to 0.3.7 (#412)

Bumps distlib from 0.3.6 to 0.3.7.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: distlib dependency-type: indirect update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``bc9f01d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bc9f01dd90688ef57f755d1b8ca5c5f7739d9d5d>`_)

- chore(deps-dev): bump pluggy from 1.0.0 to 1.2.0 (#413)

Bumps pluggy from 1.0.0 to 1.2.0.

- Changelog
- Commits

---

updated-dependencies:

- dependency-name: pluggy dependency-type: indirect update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``be8af3e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/be8af3e950d3908179e0f194132222bd04310c36>`_)

- chore(deps-dev): bump typed-ast from 1.5.4 to 1.5.5 (#411)

Bumps typed-ast from 1.5.4 to 1.5.5.

- Changelog
- Commits

---

updated-dependencies:

- dependency-name: typed-ast dependency-type: indirect update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``75302b1`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/75302b1de9ad9245327fa3b09181c7ff381fefe8>`_)

- chore(deps-dev): bump lxml from 4.9.2 to 4.9.3 (#405)

Bumps lxml from 4.9.2 to 4.9.3.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: lxml dependency-type: direct:development update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``6aa057b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6aa057bb2f0e3804e57b799fd9c3f969fb328fb7>`_)

- chore(deps-dev): bump mypy from 1.4.0 to 1.4.1 (#400)

Bumps mypy from 1.4.0 to 1.4.1.

- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``54d6a1a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/54d6a1a676d0d9715acd0d9275410b95bd9b82cf>`_)

Ci

- ci: streamline concurrency for deploy (#406)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``6a7ddfa`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6a7ddfa635995f5dbb849ba5141dcb19a70db0ea>`_)

- ci: run examples on prod-deps only (#402)
- ci: run examples on prod-deps only

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- ci: simplify ci

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``cf40048`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/cf40048f00d4d9a70306ee414ebf5a1f970c6a70>`_)

- ci: run examples (#401)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``058f386`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/058f38609453ec738d9cdaa01cbed1b22066cc77>`_)

Documentation

- docs(examples): showcase shorthand dependency management (#403)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``8b32efb`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8b32efb322a3281d58e9f980bb9001b112aa944a>`_)

Feature

- feat: programmatic access to library's version (#417)

adds `cyclonedx.__version__`

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``3585ea9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3585ea9911ae521e86793ef18f5891289fb0b604>`_)

6.1.33 v4.0.1 (2023-06-28)

Chore

- chore(deps): bump python-semantic-release/python-semantic-release (#393)

Bumps python-semantic-release/python-semantic-release from 7.33.2 to 7.34.6.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: python-semantic-release/python-semantic-release dependency-type: direct:production update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``2180d31`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2180d31e21736f535745878d2459ba6603b2b0d3>`_)

- chore(deps-dev): bump mypy from 1.3.0 to 1.4.0 (#395)
- chore(deps-dev): bump mypy from 1.3.0 to 1.4.0

Bumps mypy from 1.3.0 to 1.4.0.

- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com>

- style: ignore type confusion

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

Signed-off-by: dependabot[bot] <support@github.com> Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> Co-authored-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*ab36db4*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ab36db4a77e4a343f8699726c438e5b5233badbe>`_)

- chore(deps): bump filelock from 3.10.7 to 3.12.2 (#394)

Bumps filelock from 3.10.7 to 3.12.2.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: filelock dependency-type: indirect update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*90b339b*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/90b339b34c3afeb11d1044d9dd3fcb3feea47327>`_)

- chore(deps-dev): bump coverage from 7.2.6 to 7.2.7 (#390)

Bumps coverage from 7.2.6 to 7.2.7.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: coverage dependency-type: direct:development update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*638d472*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/638d472d474f286c3adff6e35b5ea354ef140153>`_)

- chore(deps-dev): bump xmldiff from 2.6.1 to 2.6.3 (#388)

Bumps xmldiff from 2.6.1 to 2.6.3.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: xmldiff dependency-type: direct:development update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*b5fa67c*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b5fa67c50216029af16d0643d6032e4a8bcde5e4>`_)

- chore(deps-dev): bump coverage from 7.2.5 to 7.2.6 (#387)

Bumps coverage from 7.2.5 to 7.2.6.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: coverage dependency-type: direct:development update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*c49c320*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c49c3203b3afc41e44355b403c2b495a322e4d8a>`_)

- chore(deps-dev): bump mypy from 1.2.0 to 1.3.0 (#385)

Bumps mypy from 1.2.0 to 1.3.0.

- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*bb6d8bc*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bb6d8bcdec1c10ca143396818d7605cc2f3277a6>`_)

- chore(deps-dev): bump xmldiff from 2.5 to 2.6.1 (#375)

Bumps xmldiff from 2.5 to 2.6.1.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: xmldiff dependency-type: direct:development update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*27b9ec5*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/27b9ec57a48bcb0c29499df8e915b956c7b06b50>`_)

- chore(deps-dev): bump mypy from 1.1.1 to 1.2.0 (#372)

Bumps mypy from 1.1.1 to 1.2.0.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*5e5a8c2*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5e5a8c25979dc0769048d36abba5b1623b797f2e>`_)

- chore(deps-dev): bump coverage from 7.2.2 to 7.2.5 (#383)

Bumps coverage from 7.2.2 to 7.2.5.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: coverage dependency-type: direct:development update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*b288d94*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b288d9406ff592c1f12be82746ccf7fd527413d7>`_)

- chore(deps): update poetry and other dependency versions (#369)
- update packageurl type hints

Signed-off-by: gruebel <anton.gruebel@gmail.com>

- lower bound packageurl-python dependency

Signed-off-by: gruebel <anton.gruebel@gmail.com>

- update deps.lowest.r

Signed-off-by: gruebel <anton.gruebel@gmail.com>

Signed-off-by: gruebel <anton.gruebel@gmail.com> (``*aa5b936*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/aa5b936f17c5a9840a0f436b8d4540439cf4c0a5>`_)

- chore: CI/QA/Build meintenance (#358)
- build: streamlined ci and builds

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- chore: upgrade lockfile with poetry1.4

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- removed extra brace

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- fixed long line

Signed-off-by: Paul Horton <paul.horton@owasp.org>

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> Signed-off-by: Paul Horton <paul.horton@owasp.org> Co-authored-by: Paul Horton <paul.horton@owasp.org> (``*9779af0*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9779af02f5f3cd99fe3e1a088f5547f4991b05b7>`_)

- chore: followup of #340 (#360)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*723ae8e*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/723ae8e4ddbffc4851c10f64692e7265973ef730>`_)

- chore: prevent dev-lowest-lockfile from dependency bumps (#359)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*16870f4*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/16870f4119865b549172cc76588ca1aa7ce00357>`_)

- chore: manually craft more accurate CHANGELOG for `4.0.0`

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``*32ce3a2*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/32ce3a2ca018b8afcfcb101cad8fac80c547ddc5>`_)

Ci

- ci: cannot use variables in `uses`

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``*2371a1b*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2371a1bdc39c85ee65e43ac8bb22cae1b199385e>`_)

- ci: cannot use variables in `uses`

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``*aa0eab1*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/aa0eab134c85e7501134f8a417c34e430abc7101>`_)

- ci: add concurrency rules (#361)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*f65d646*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f65d64699a48bd6fe540c7503491ce29b1ce38d1>`_)

Documentation

- docs(examples): README (#399)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*1d262ba*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1d262ba57eab0d61b947fc293fc59c6234f19647>`_)

- docs: add exaple how to build and serialize (#397)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*65e22bd*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/65e22bdc6a1a3fc02a6282146bc8fbc17ddb32fa>`_)

Fix

- fix: conditional warning if no root dependencies were found (#398)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``c8175bb`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c8175bb6aebac7f129d42d7a5a0ae928212c20cb>`_)

Unknown

- 4.0.1

Automatically generated by python-semantic-release (``4a72f51`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/4a72f515ad7b5e46a07f31bea18a94b162e87715>`_)

- Add missing space in warning message. (#364)

Signed-off-by: Michael Schlenker <michael.schlenker@contact-software.com> Co-authored-by: Michael Schlenker <michael.schlenker@contact-software.com> (``dad0d28`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/dad0d28ceb7381d1b503e5b29776fc01513f8b04>`_)

6.1.34 v4.0.0 (2023-03-20)

Breaking

- feat: Release 4.0.0 #341)

Highlights of this release include:

- Support for De-serialization from JSON and XML to this Pythonic Model
- Deprecation of Python 3.6 support
- Support for Python 3.11
- Support for `BomLink`
- Support VEX without needing `Component` in the same `Bom`
- Support for `services` having `dependencies`

BREAKING CHANGE: Large portions of this library have been re-written for this release and many methods and contracts have changed.

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- feat: support VEX without Components in the same BOM

BREAKING CHANGE: Model classes changed to relocated Vulnerability at Bom, not at Component

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- feat: support VEX without Components in the same BOM

BREAKING CHANGE: Model classes changed to relocated Vulnerability at Bom, not at Component

Signed-off-by: Paul Horton <paul.horton@owasp.org>

feat: allow `version` of BOM to be defined

feat: allow `serial_number` of BOM to be prescribed

feat: add helper method to get URN for a BOM according to https://www.iana.org/assignments/urn-formal/cdx Signed-off-by: Paul Horton <paul.horton@owasp.org>

- chore: fix release workflow
- chore: editorconfig

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- feat: support for deserialization from JSON and XML (#290)

BREAKING CHANGE:

- feat: drop Python 3.6 support

Signed-off-by: Hakan Dilek <hakandilek@gmail.com> Signed-off-by: Paul Horton <paul.horton@owasp.org> Co-authored-by: Hakan Dilek <hakandilek@gmail.com> Co-authored-by: Hakan Dilek <hakandilek@users.noreply.github.com>

- fix: update `serializable` to include XML safety changes

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- feat: Support for Python 3.11 (#349)
- feat: officially test and support Python 3.11

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- removed unused imports

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- bump `poetry` to `1.1.12` in CI

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- fix: remove `toml` as dependency as not used and seems to be breaking Python 3.11 CI

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- fix: removed `types-toml` from dependencies - not used

Signed-off-by: Paul Horton <paul.horton@owasp.org>

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- fix: removed `autopep8` in favour of `flake8` as both have conflicting dependencies now

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- chore: bump dev dependencies

fix: removed `setuptools` as dependency Signed-off-by: Paul Horton <paul.horton@owasp.org>

- tests: compoennt versions optional (#350)
- chore: exclude `venv*` from QA; add typing to QA

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- tests: component versions are optional

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- doc: doc updates for new deserialization feature

Signed-off-by: Paul Horton <paul.horton@owasp.org>

- doc: doc updates for contribution

Signed-off-by: Paul Horton <paul.horton@owasp.org>

Signed-off-by: Paul Horton <paul.horton@owasp.org> Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> Signed-off-by: Hakan Dilek <hakandilek@gmail.com> Co-authored-by: Jan Kowalleck <jan.kowalleck@gmail.com> Co-authored-by: Hakan Dilek <hakandilek@gmail.com> Co-authored-by: Hakan Dilek <hakandilek@users.noreply.github.com> (``*8fb1b14*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8fb1b14f5e04e85f21e654c44fa6b9b774867757>`_)

Chore

- chore(deps): bump relekang/python-semantic-release from 7.31.2 to 7.33.1 (#345)

Bumps relekang/python-semantic-release from 7.31.2 to 7.33.1.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: relekang/python-semantic-release dependency-type: direct:production update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*a011d89*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a011d89ce6cee9e56bcfcc9a9338fa1e559721f7>`_)

- chore: package manifest fix link to homepage and documentation (#291)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*f2350b4*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f2350b4e2b0fb7668ca987e523c53acb6ac6fefb>`_)

Unknown

- 4.0.0

Automatically generated by python-semantic-release (``*40fbfda*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/40fbfda428cfa71b16fd6e5e8d5f49cea4b5438b>`_)

6.1.35 v3.1.5 (2023-01-12)

Chore

- chore: do not ship exra LICENSE file (#339)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*b7f1028*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b7f1028156de8d1e14a391d84d24aa697814902a>`_)

Fix

- fix: mak test's schema paths relative to `cyclonedx` package (#338)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*1f0c05f*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1f0c05fe2b2a22bc84a1a437dd59390f2ceaf986>`_)

Unknown

- 3.1.5

Automatically generated by python-semantic-release (``*ba603cf*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ba603cf96fad51a85d5159e83c402d613fefbb7c>`_)

6.1.36 v3.1.4 (2023-01-11)

Chore

- chore: add Jan Kowalleck as a maintainer

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*7aae26d*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7aae26d09c8c0d6976f10d94c2bfbd4cb8f11a0b>`_)

Fix

- fix(tests): include tests in `sdist` builds (#337)
- feat: include `tests` in `sdist` builds for #336
- delete unexpected `DS_Store` file

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*936ad7d*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/936ad7d0c26d8f98040203d3234ca8f1afbd73ab>`_)

Test

- test: mock `ThisTool.version` for constisten results (#335)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*57a9e5e*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/57a9e5e4f5b1eb785984be9d5a35aac60315232d>`_)

Unknown

- 3.1.4

Automatically generated by python-semantic-release (``*0b19294*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0b19294e4820f0da5e81decd4d902ef7789ecb61>`_)

6.1.37 v3.1.3 (2023-01-07)

Fix

- fix: serialize dependency graph for nested components (#329)
- tests: regression tests for issue #328
- fix: for issue #328

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``fb3f835`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fb3f8351881783281f8b7e796098a4c145b35927>`_)

Test

- test: tidy up test beds (#333)
- test: consolidate imports
- test: recreate all fixtures
- test: docs

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``ab862e7`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ab862e79b72b808693e2ec7f6fe1fa3e99cae011>`_)

Unknown

- 3.1.3

Automatically generated by python-semantic-release (``11a420c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/11a420c5fc38bb48d2a91713cc74574acb131184>`_)

6.1.38 v3.1.2 (2023-01-06)

Chore

- chore(deps): bump Gr1N/setup-poetry from 7 to 8 (#326)

Bumps Gr1N/setup-poetry from 7 to 8.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: Gr1N/setup-poetry dependency-type: direct:production update-type: version-update:semver-major ...

Signed-off-by: dependabot[bot] <support@github.com>

Signed-off-by: dependabot[bot] <support@github.com> Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``f3af229`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f3af22979978f0c38c4c8f48b4271ee6a6c1e1bd>`_)

- chore: editorconfig

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``8c75b1b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8c75b1ba63c10929c005ea27ebb6f63afa8b9719>`_)

Ci

- ci: fix py36 (#320)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``cf9f790`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/cf9f790e30f5b430ea1ece8916b54323e1cdb5ee>`_)

Documentation

- docs: typo

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``539b57a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/539b57a00e4e60e239bb26141f219366121e7bc2>`_)

- docs: fix shields (#324)

caused by https://github.com/badges/shields/issues/8671

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``555dad4`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/555dad4bc255066036ecca028192eb83df8ba5a0>`_)

- docs: fix typo (#318)

Signed-off-by: Roland Weber <rolweber@de.ibm.com> (``63bfb87`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/63bfb8772fe78e9842675d17862c456150dbbc15>`_)

Fix

- fix: prevent errors on metadata handling for some specification versions (#330)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``f08a656`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f08a65649aee750397edc061eb3b8325a69bb4b4>`_)

Style

- style: split joined path segments (#331)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``493104c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/493104c1bccc669ee55b89a2c360268d36f3f1b7>`_)

Unknown

- 3.1.2

Automatically generated by python-semantic-release (``0853d14`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0853d14780b8e44e9b285bee2ac6b81551640c5f>`_)

- clarify sign-off step (#319)

Signed-off-by: Roland Weber <rolweber@de.ibm.com> (``007fb96`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/007fb96a1ec23b9516bc383afa85b3efc2707aa8>`_)

6.1.39 v3.1.1 (2022-11-28)

Chore

- chore: CHANGELOG typos (``6c0c174`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6c0c1742d2ea19dfc0284785cf9597b43ef05979>`_)
- chore: update CHANGELOG to explain jump from `2.7.1` to `3.1.0`. (``1b8cd12`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1b8cd12b03adb03451ed8ee4562161bd82a18972>`_)

Fix

- fix: type hint for `get_component_by_purl` is incorrect

chore: force automated release Signed-off-by: Paul Horton <paul.horton@owasp.org> (``3f20bf0`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3f20bf04a65d5c539230281437255b5f48e17621>`_)

Unknown

- 3.1.1

Automatically generated by python-semantic-release (``503955e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/503955ea9e19e1d3ca611df36508dcf1aa93905c>`_)

- Merge pull request #310 from gruebel/fix-method-type-hint

fix: type hint for `get_component_by_purl` is incorrect (``06037b9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/06037b99e0d6ebc5388d3c5e0799a68233ed92e8>`_)

- move tests to model bom file

Signed-off-by: gruebel <anton.gruebel@gmail.com> (``4c8a3ab`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/4c8a3ab0eef349c007285ff9dfed0c00c6732a96>`_)

- fix type hint for get_component_by_purl

Signed-off-by: gruebel <anton.gruebel@gmail.com> (``735c05e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/735c05eebb792eed55aeb4d5a7be8043ee1cd9ae>`_)

6.1.40 v3.1.0 (2022-09-15)

Chore

- chore: fix release workflow (``5863622`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/586362272af3f5fd7a11c1c65502bca31d8813eb>`_)
- chore: fix poetry in tox

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``7f8c668`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7f8c668cf152af554dbc5183f275723cd3d472b2>`_)

Feature

- feat: out-factor SPDX compund detection

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``fd4d537`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fd4d537c9dced0e38f14d99dee174cc5bb0bd465>`_)

- feat: out-factor SPDX compund detection

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``2b69925`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2b699252f8857d97231a689ea9cbfcdff9459626>`_)

- feat: license factories

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``033bad2`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/033bad2a50fd2236c712d4621caa57b04fcc2043>`_)

Test

- test: license factories

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``baf83f9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/baf83f9aebe4cdf38341c2432bf8a97e74db5105>`_)

Unknown

- 3.1.0

Automatically generated by python-semantic-release (``e52c174`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e52c17447b1520103ccb24192ab92560429df595>`_)

- Merge pull request #305 from CycloneDX/license-factories

feat: add license factories to more easily support creation of `License` or `LicenseChoice` from SPDX license strings #304 (``5ff4494`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5ff4494b0e0d76d04cf8a4245ce0426f0abbd8f9>`_)

- tests: refactor tests

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``3644f13`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3644f1357ae6b0e1f84e442cd6d9a045fc26fbce>`_)

- tests: rebase/fixup poetry lock

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``26817c0`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/26817c0089bfd4083ecfb5ce85039c8d75b84606>`_)

- Merge pull request #301 from CycloneDX/fix-poetry-in-tox

chore: fix poetry in tox (``92aea8d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/92aea8d3413cd2af820cc8160ef48a737951b0ea>`_)

- remove v3 from CHANGELOG #286 (#287)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``7029721`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/702972105364a3ab225ea5a586c48cec664601ca>`_)

- 3.0.0

Automatically generated by python-semantic-release (``69582ff`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/69582ff7a9e3a1cfb2c7193c3d194d69e35899c1>`_)

6.1.41 v2.7.1 (2022-08-01)

Chore

- chore: manual fix release publication `2.7.1`

Signed-off-by: Paul Horton <paul.horton@owasp.org> (*`b569548`* <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b56954840ada89c0ba63b4be16e099cd74cc001d>`_)

- chore(deps-dev): bump flake8-isort from 4.1.1 to 4.1.2.post0 (#280)

Bumps flake8-isort from 4.1.1 to 4.1.2.post0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: flake8-isort dependency-type: direct:development update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (*`01cb53b`* <https://github.com/CycloneDX/cyclonedx-python-lib/commit/01cb53b9a29f0dfa35b57d4ac0ac56f2d8778f0a>`_)

- chore: resolve hang issue with running isort as pre-commit hook

Signed-off-by: Paul Horton <paul.horton@owasp.org> (*`fb25b70`* <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fb25b70c0a3b5a5855332e1c5371219b97beb181>`_)

- chore: re-added `isort` to pre-commit hooks ran isort

Signed-off-by: Paul Horton <paul.horton@owasp.org> (*`051e543`* <https://github.com/CycloneDX/cyclonedx-python-lib/commit/051e5436fc5d317286d0d25c8987cf236d20af08>`_)

Ci

- ci: change pinned version of python-semantic-release as preventing automated releases

Signed-off-by: Paul Horton <paul.horton@owasp.org> (*`6e12be7`* <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6e12be70fb2a71de60428155b4d0ae82fa43ef2d>`_)

Fix

- fix: pinned `mypy &lt;= 0.961` due to #278

Signed-off-by: Paul Horton <paul.horton@owasp.org> (*`d6955cb`* <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d6955cb86d8da7a72d0146d0dbeb7c34a794a954>`_)

- fix: properly support nested `components` and `services` #275

Signed-off-by: Paul Horton <paul.horton@owasp.org> (*`6597db7`* <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6597db740f222c68ad90f74fb8fdb58b72642adb>`_)

Unknown

- Merge pull request #276 from CycloneDX/fix/bom-validation-nested-components-isue-275

fix: BOM validation fails when Components or Services are nested #275

fix: updated dependencies #271, #270, #269 and #256 (``68a0cdd`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/68a0cddc0a226947d76b6a275cfceba383797d3b>`_)

- Merge branch 'main' into fix/bom-validation-nested-components-isue-275 (``6caee65`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6caee657260e46f18cade24a73b4f17bc5ad6dd8>`_)
- added tests to cover new `Component.get_all_nested_components()` method

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``75a77ed`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/75a77ed6576f362435d1a3e6e59cbc5d871b9971>`_)

- Revert "chore: re-added `isort` to pre-commit hooks"

This reverts commit f50ee1eb79f3f4e5b9d21824e64192d0af43d3f0.

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``5f7f30e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5f7f30e6a79f7cef6fff296ae0d7e5381f9b5cda>`_)

- removed tests where services are part of dependency tree - see #277

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``f26862b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f26862b0b7f85e3610efbdf17cf304ddc71e5366>`_)

- aded XML output tests for Issue #275

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``ebef5f2`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ebef5f212fec13fc8c9bf00553f9bf3f77a0d3f6>`_)

- updated XML output tests

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``356c37e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/356c37ebea85eb10e2505f2b16264d95f292bd55>`_)

- addressed JSON output for #275 including test addiitions

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``692c005`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/692c005c686157134a79e3ffc8ab1e7ce8942de9>`_)

6.1.42 v2.7.0 (2022-07-21)

Chore

- chore(deps): bump virtualenv from 20.15.0 to 20.15.1 (#255)

Bumps virtualenv from 20.15.0 to 20.15.1.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: virtualenv dependency-type: indirect update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*d720a5f*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d720a5fed662eaf19657d5a2d3f46a9b386d13de>`_)

- chore(deps-dev): bump flake8-bugbear from 22.6.22 to 22.7.1 (#259)

Bumps flake8-bugbear from 22.6.22 to 22.7.1.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: flake8-bugbear dependency-type: direct:development update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*1175f60*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1175f603f863bbcdb3d49dd84c66a25a5826c6ea>`_)

- chore(deps-dev): bump jsonschema from 4.6.0 to 4.6.1 (#258)

Bumps jsonschema from 4.6.0 to 4.6.1.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: jsonschema dependency-type: direct:development update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*ddbfabc*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ddbfabce2487f21ef204674dc5bd8de70c8fd204>`_)

- chore(deps-dev): bump lxml from 4.9.0 to 4.9.1 (#257)

Bumps lxml from 4.9.0 to 4.9.1.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: lxml dependency-type: direct:development update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*f045b7f*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f045b7ffcf318652dd8a13b7fe5c61f3b4d81a7b>`_)

- chore(deps): bump virtualenv from 20.14.1 to 20.15.0 (#251)

Bumps virtualenv from 20.14.1 to 20.15.0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: virtualenv dependency-type: indirect update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``70270a9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/70270a97b481d976eea82bd3c35bbb5055104234>`_)

- chore(deps-dev): bump flake8-bugbear from 22.4.25 to 22.6.22 (#252)

Bumps flake8-bugbear from 22.4.25 to 22.6.22.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: flake8-bugbear dependency-type: direct:development update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``c957226`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c957226543b43631d247f3417621668cc824232a>`_)

Feature

- feat: support for CycloneDX schema `1.4.2` - adds `vulnerability.properties` to the schema (``32e7929`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/32e792928bdf37133e966ef72ec01b0bc698482d>`_)
- feat: support for CycloneDX schema version `1.4.2`
- Provides support for `vulnerability.properties`

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``db7445c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/db7445cd343fc35c6d6fc9f5af3e28cf97a19732>`_)

- feat: added updated CycloneDX 1.4.2 schemas

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``7fb27ae`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7fb27aed58f7de10f8c6b703699bba315af353e7>`_)

Unknown

- 2.7.0

Automatically generated by python-semantic-release (``96d155e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/96d155e864d83482242c22f69af8e7c618d05a1b>`_)

6.1.43 v2.6.0 (2022-06-20)

Chore

- chore(deps): bump colorama from 0.4.4 to 0.4.5 (#249)

Bumps colorama from 0.4.4 to 0.4.5.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: colorama dependency-type: indirect update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``39637ad`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/39637ade2668003c3bf7c22cf40c72bae324d8c1>`_)

Feature

- feat: reduce unnessessarry type casting of `set`/`SortedSet` (#203)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``089d971`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/089d9714f8f9f8c70076e48baa18340899cc29fa>`_)

Unknown

- 2.6.0

Automatically generated by python-semantic-release (``8481e9b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8481e9bd8dc5196c2e703e5cd19974bb22bc270e>`_)

6.1.44 v2.5.2 (2022-06-15)

Chore

- chore(deps): bump actions/setup-python from 3 to 4 (#247)

Bumps actions/setup-python from 3 to 4.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: actions/setup-python dependency-type: direct:production update-type: version-update:semver-major …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*ddd0144*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ddd01446e5fe201bfb0cebeee3c4afb25f54223b>`_)

Fix

- fix: add expected lower-than comparators for `OrganizationalEntity` and `VulnerabilityCredits` (#248)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*0046ee1*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0046ee19547be8dafe5d73bad886b9c5f725f26e>`_)

Unknown

- 2.5.2

Automatically generated by python-semantic-release (``*fb9a796*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fb9a796d0b34c2d930503790c74d6d7ed5e3c3d6>`_)

6.1.45 v2.5.1 (2022-06-10)

Chore

- chore(deps-dev): bump mypy from 0.960 to 0.961 (#244)

Bumps mypy from 0.960 to 0.961.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*48ea951*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/48ea951c92f0b944e5aae2cd1cfd299b02fb4322>`_)

Fix

- fix: add missing `Vulnerability` comparator for sorting (#246)

Partial fix for #245.

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``c3f3d0d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c3f3d0d105f0dcf991175040b6d6c2b6e7e25d8f>`_)

Unknown

- 2.5.1

Automatically generated by python-semantic-release (``1ea5b20`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1ea5b20f1c93e6e6b3799444c7ea6fd65a2e068c>`_)

6.1.46 v2.5.0 (2022-06-10)

Build

- build: move typing to dev-dependencies

Move `types-setuptools` and `types-toml` to dev-dependencies (#226)

Signed-off-by: Adam Johnson <me@adamj.eu> (``0e2376b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0e2376baade068ae0490b05550837d104e9abfa4>`_)

Chore

- chore(deps-dev): bump jsonschema from 4.5.1 to 4.6.0 (#242)

Bumps jsonschema from 4.5.1 to 4.6.0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: jsonschema dependency-type: direct:development update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``32af991`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/32af991c8f69c7f9f2f06b68c014bc7af0498d5d>`_)

- chore(deps-dev): bump lxml from 4.8.0 to 4.9.0 (#241)

Bumps lxml from 4.8.0 to 4.9.0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: lxml dependency-type: direct:development update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*6d5189e*`<https://github.com/CycloneDX/cyclonedx-python-lib/commit/6d5189e4612126a2fcc72ffe77857ab6fbea25bc>`_)

- chore(deps-dev): bump mypy from 0.942 to 0.960 (#230)

Bumps mypy from 0.942 to 0.960.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*88d9d8b*`<https://github.com/CycloneDX/cyclonedx-python-lib/commit/88d9d8b7ff18f495a0767e3ed9f37783030ca45d>`_)

- chore(deps): bump types-setuptools from 57.4.12 to 57.4.17 (#238)

Bumps types-setuptools from 57.4.12 to 57.4.17.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: types-setuptools dependency-type: direct:production update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*3d011ab*`<https://github.com/CycloneDX/cyclonedx-python-lib/commit/3d011ab8f46a3486e1f0dc2a4bb099f7e68f31dd>`_)

- chore(deps): bump types-setuptools from 57.4.12 to 57.4.17 (#237)

Bumps types-setuptools from 57.4.12 to 57.4.17.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: types-setuptools dependency-type: direct:production update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*a1d1bae*`<https://github.com/CycloneDX/cyclonedx-python-lib/commit/a1d1bae1e5a1e3fdabba3082b3f1a94e3265312d>`_)

- chore(deps): bump typed-ast from 1.5.2 to 1.5.4 (#232)

Bumps typed-ast from 1.5.2 to 1.5.4.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: typed-ast dependency-type: indirect update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*866f9ac*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/866f9ac4e4f270fd24b04766aa0082dac6116359>`_)

- chore(deps-dev): bump jsonschema from 4.4.0 to 4.5.1 (#221)

Bumps jsonschema from 4.4.0 to 4.5.1.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: jsonschema dependency-type: direct:development update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*c65ce28*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c65ce284d602b9218464cc8b2cfbcff6b13aa910>`_)

Ci

- ci: fix run with lowest compat dependencies (#240)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*a4596c8*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a4596c8023553a15e33b45e84142e4ef27591b6a>`_)

- ci: pin GH-action `semantic-release` to v7.28.1 (#234)

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*91e1297*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/91e12971bf90fffb5b440b2acc74a3f8614932bd>`_)

Documentation

- docs: fix typo "This is out" -> "This is our"

Fix typo in comments: "This is out" -> "This is our" (#233)

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``ef0278a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ef0278a2044147e73a281c5a59f95049d4af7641>`_)

Feature

- feat: use `SortedSet` in model to improve reproducibility - this will provide predictable ordering of various items in generated CycloneDX documents - thanks to @RodneyRichardson

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``8a1c404`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8a1c4043f502292b32c4ab36a8618cf3f67ac8df>`_)

Test

- test: tests calculate versions if needed

Don't hardcode component version in test (#229)

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``7b3ce65`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7b3ce65f92ff6009a1e29d4938eac5ea664b2538>`_)

Unknown

- 2.5.0

Automatically generated by python-semantic-release (``c820423`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c820423ffffb90ec7a42d8873d99428277f9ae28>`_)

- Merge pull request #235 from RodneyRichardson/use-sorted-set

feat: use `SortedSet` in model to improve reproducibility - this will provide predictable ordering of various items in generated CycloneDX documents - thanks to @RodneyRichardson (``c43f6d8`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c43f6d8ce41a9de91a84cea7a40045cab8121792>`_)

- Merge branch 'CycloneDX:main' into use-sorted-set (``1b8ac25`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1b8ac252a28af1b938d6cad4182e6f2d586b26c0>`_)

- Fix SortedSet type hints for python < 3.8

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``71eeb4a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/71eeb4aeeb9e911df2422c097ebfb671c648242d>`_)

- Fix line length warning.

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``e9ee712`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e9ee71291da882a924a9edec7d1f5d6be62797e6>`_)

- Fix more type hints for python < 3.8

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``f042bce`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f042bcef1829a852dd787e226d883f5bbd5c39c3>`_)

- Fix SortedSet type hints for python < 3.8

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``2e283ab`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2e283abed0b67e9e70c825e0d7c6ad7e6691c678>`_)

- Fix type hint on ComparableTuple

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``43ef908`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/43ef908d61fd03e5a4c2ecfabdf22764c8613429>`_)

- Sort usings.

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``8f86c12`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8f86c1292d5d0c550a4ec6018b81400255567f93>`_)

- Fix sonatype-lift warnings

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``f1e92e3`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f1e92e3cfbe9df2b07b745582608f9f72531684c>`_)

- Fix warnings.

Change tuple -> Tuple Fix Diff initialization Add sorting to AttachedText

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``2b47ff6`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2b47ff612335b538ceab5e77b60dbe058f739e2e>`_)

- Reduce sortedcontainers.pyi to only the functions used.

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``ef0fbe2`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ef0fbe2130f763888cb34e8e71a6520d282a0cda>`_)

- Remove flake8 warnings

Remove unused imports and trailing whitespace. Sort usings in pyi file.

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``41d1bee`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/41d1bee824381c25a8c6870abeb1f484c33c78ba>`_)

- Add type hints for SortedSet

Fix use of set/Set.

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``df0f554`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/df0f554bff311886705327fd863d573e82123f9e>`_)

- Replace object type hint in **lt** with Any

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``ec22f68`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ec22f683e1b12843421a23cff15f91628a7dfffe>`_)

- Make reorder() return type explicit List (as flagged by sonatype-lift bot)

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``695ee86`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/695ee862ce9043807a9d825324970cd1b770a46c>`_)

- Use SortedSet in model to improve reproducibility

Added `__lt__()` to all model classes used in SortedSet, with tests Explicitly declared Enums as (str, Enum) to allow sorting Added dependency to sortedcollections package

Signed-off-by: Rodney Richardson <rodney.richardson@cambridgeconsultants.com> (``368f522`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/368f5221e54a635cd03255efd56d4da2a8d7f56b>`_)

6.1.47 v2.4.0 (2022-05-17)

Chore

- chore(deps): bump virtualenv from 20.14.0 to 20.14.1 (#208)

Bumps virtualenv from 20.14.0 to 20.14.1.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: virtualenv dependency-type: indirect update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*04f3671*`<https://github.com/CycloneDX/cyclonedx-python-lib/commit/04f3671de036b340faf18170603fad32095771cb>`_)

- chore(deps-dev): bump tox from 3.24.5 to 3.25.0 (#209)

Bumps tox from 3.24.5 to 3.25.0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: tox dependency-type: direct:development update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*8eee5d3*`<https://github.com/CycloneDX/cyclonedx-python-lib/commit/8eee5d354c3ee640bbc773d315f1c17e1a8334fd>`_)

- chore(deps): bump types-toml from 0.10.4 to 0.10.7 (#222)

Bumps types-toml from 0.10.4 to 0.10.7.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: types-toml dependency-type: direct:production update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*5d19805*`<https://github.com/CycloneDX/cyclonedx-python-lib/commit/5d19805c4e0568d4fc0894ed0b9d7cb3b99e219b>`_)

- chore(deps-dev): bump flake8-bugbear from 22.3.23 to 22.4.25 (#220)

Bumps flake8-bugbear from 22.3.23 to 22.4.25.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: flake8-bugbear dependency-type: direct:development update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``de7f4aa`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/de7f4aae0378c6475d65ac9ec2303155d4062591>`_)

Feature

- feat(deps): remove unused `typing-extensions` constraints

PullRequest and details via #224

Signed-off-by: gruebel <anton.gruebel@gmail.com> (``2ce358a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2ce358a37e6ce5f06aa9297aed17f8f5bea38e93>`_)

Unknown

- 2.4.0

Automatically generated by python-semantic-release (``4874354`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/48743542fd2f3219a4f2295f363ae6e5bcf2a738>`_)

- revert `types-toml` on lowest setup (``32ece98`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/32ece98b24fd6966722b8cdf698f01b8fb1b8821>`_)

6.1.48 v2.3.0 (2022-04-20)

Feature

- feat: add support for Dependency Graph in Model and output serialisation

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``ea34513`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ea34513f8229a909007793288ace2f6f51684333>`_)

Unknown

- 2.3.0

Automatically generated by python-semantic-release (``5c1047a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5c1047afc75726cca4130b90b8459418ec6342e8>`_)

- Merge pull request #210 from CycloneDX/feat/support-bom-dependencies

feat: add support for Dependency Graph in Model and output serialisation (JSON and XML) (``938169c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/938169c05b458967cd1dabc338981d296f5b2842>`_)

- Merge pull request #214 from CycloneDX/feat/support-bom-dependencies-no-cast

no cast (``2551545` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/25515456f2707964032c1f9642bae3d79ba2b994>`_)

- no cast

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``dec3b70` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/dec3b703f7e69cd2b3fdff34583ee052b1cbb1d2>`_)

- update to use `Set` operators (more Pythonic)

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``f01665e` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f01665e96c87b9dd1fdb37d907a8339ba819e2cc>`_)

- missing closing `>` in `BomRef.__repr__`

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``2c7c4be` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2c7c4be8210231dcfaf9e8937bd943f3ea6683c3>`_)

- removed unnecessary condition - `self.get_bom().components` is always a `Set`

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``5eb5669` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5eb5669bdeb982c9f0b4a72f2264a8559e9a3bc3>`_)

- added additional tests to validate Component in Metadata is properly represented in Dependency Graph

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``b8d526e` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b8d526ee52b3923c7755a897e0c042c159fb8d99>`_)

- adjusted unit tests to account for inclusion of Component in Bom Metadata in Dependency Graphy

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``c605f2b` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c605f2be90092f09bb0eb89dccb27767d78dcfac>`_)

- updates based on feedback from @jkowalleck

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``04511f3` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/04511f3c523bc26b0b434d8334d37eccaaaf1ea4>`_)

- Merge branch 'feat/support-bom-dependencies' of github.com:CycloneDX/cyclonedx-python-lib into feat/support-bom-dependencies (``8fb408c` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8fb408cfe7941efca424777a94084755ee8a50e4>`_)

- doc: updated docs to reflect support for Dependency Graph

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``a680544` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a68054491529631c792e51c764bbf64a5e9b4834>`_)

- updated file hash in test

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``56f3d5d` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/56f3d5d432b6c50679cfd733cf2b0ed2ea55400e>`_)

- removed unused import

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``61c3338` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/61c3338e139a8e1a72a659080f2043b352007561>`_)

- doc: updated docs to reflect support for Dependency Graph

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``3df017f` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3df017feaaa461bcfa7082f58a5824aa92493b59>`_)

- updated file hash in test

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``449cb1e` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/449cb1e56e64e6c144c0d2b6b69649df2d6e5320>`_)

- removed unused import

Signed-off-by: Paul Horton <paul.horton@owasp.org> (``f487c4a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f487c4a44f5604fa3d1da2c0bc57d09e22057973>`_)

6.1.49 v2.2.0 (2022-04-12)

Chore

- chore(deps): bump actions/upload-artifact from 2 to 3 (#204)

Bumps actions/upload-artifact from 2 to 3.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: actions/upload-artifact dependency-type: direct:production update-type: version-update:semver-major ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``dad8538`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/dad8538797352e1f2d0bb322b2df007370da19be>`_)

- chore(deps): bump types-setuptools from 57.4.11 to 57.4.12 (#205)

Bumps types-setuptools from 57.4.11 to 57.4.12.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: types-setuptools dependency-type: direct:production update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``eae598a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/eae598adca14eaa7125ab8bc6a2af4b213cdbd5c>`_)

Ci

- ci: introduce `timeout-minutes` and drop `dependabot` branches for CI #206

Signed-off-by: Paul Horton <phorton@sonatype.com> (``e5b426f`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e5b426f0287e75f8c9c2b0937cebaab13dc069a5>`_)

Feature

- feat: Bump XML schemas to latest fix version for 1.2-1.4 - see: https://github.com/CycloneDX/specification/issues/122

Signed-off-by: Paul Horton <phorton@sonatype.com> (``bd2e756`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bd2e756de15c37b34d2866e8de521556420bd5d3>`_)

- feat: bump JSON schemas to latest fix verison for 1.2 and 1.3 - see:
- https://github.com/CycloneDX/specification/issues/123
- https://github.com/CycloneDX/specification/issues/84
- https://github.com/CycloneDX/specification/issues/125

Signed-off-by: Paul Horton <phorton@sonatype.com> (``bd6a088`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bd6a088d51c995c0f08271f56aedb456c60c1a2e>`_)

Unknown

- 2.2.0

Automatically generated by python-semantic-release (``67ecfac`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/67ecfacc38817398319ac5d627f2b3a17fb45b3f>`_)

- Merge pull request #207 from CycloneDX/feat/update-schemas

feat: Update CycloneDX Schemas to latest patch versions (``2c55cb5`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2c55cb51042694d48a2eccd8e505833196effb59>`_)

- mark schema files as vendored

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``a9c3e77`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a9c3e77998e7c05af5ba097891cd05a8cdb89232>`_)

- Merge pull request #191 from CycloneDX/feat/pre-commit-hooks

[DEV] Add pre-commit hooks (``91ceeb1`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/91ceeb1fdafddf20af546d383a2fb16393977ef5>`_)

6.1.50 v2.1.1 (2022-04-05)

Chore

- chore: shield icons in README (``87c490e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/87c490e883f1c68a96233ca6d83e641481fb83a4>`_)

Fix

- fix: prevent error if `version` not set

Signed-off-by: Paul Horton <phorton@sonatype.com> (``b9a84b5`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b9a84b5b39fe6cb1560764e86f8bd144f2a901e3>`_)

Unknown

- 2.1.1

Automatically generated by python-semantic-release (``f78d608`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f78d6081abc1a8adb80ef0c79a07c624ad9e3a5c>`_)

- Merge pull request #194 from CycloneDX/fix/json-output-version-optional-bug-193

fix: `version` being optional in JSON output can raise error (``6f7e09a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6f7e09aa4d05a4a2dc60569732f6b2ae5582a154>`_)

6.1.51 v2.1.0 (2022-03-28)

Chore

- chore(deps): bump virtualenv from 20.13.4 to 20.14.0 (#200)

Bumps virtualenv from 20.13.4 to 20.14.0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: virtualenv dependency-type: indirect update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``6ccb637`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6ccb63789fdc49c2b0b7f1349f4a4f168951ed73>`_)

- chore(deps-dev): bump mypy from 0.941 to 0.942 (#199)

Bumps mypy from 0.941 to 0.942.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``51dadb9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/51dadb9ded4a49a9ad6e22dd689cbfbbe04547aa>`_)

- chore(deps-dev): bump flake8-bugbear from 22.1.11 to 22.3.23 (#201)

Bumps flake8-bugbear from 22.1.11 to 22.3.23.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: flake8-bugbear dependency-type: direct:development update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*4f9f169*`<https://github.com/CycloneDX/cyclonedx-python-lib/commit/4f9f1693950caecdd6b01c25c2b47c7940f703b5>`_)

- chore(deps): bump types-setuptools from 57.4.10 to 57.4.11 (#197)

Bumps types-setuptools from 57.4.10 to 57.4.11.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: types-setuptools dependency-type: direct:production update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*8f4db6b*`<https://github.com/CycloneDX/cyclonedx-python-lib/commit/8f4db6b99b1213949c69605019e468ca9598a8e0>`_)

- chore(deps-dev): bump mypy from 0.940 to 0.941 (#195)

Bumps mypy from 0.940 to 0.941.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*8012c29*`<https://github.com/CycloneDX/cyclonedx-python-lib/commit/8012c299634537340a061e9b1b3ad60071fd7c13>`_)

- chore(deps): bump virtualenv from 20.13.3 to 20.13.4 (#196)

Bumps virtualenv from 20.13.3 to 20.13.4.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: virtualenv dependency-type: indirect update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*f94bb64*`<https://github.com/CycloneDX/cyclonedx-python-lib/commit/f94bb64f5216eb8de8f032368e3c73f914e0b737>`_)

- chore(deps): bump testfixtures from 6.18.4 to 6.18.5 (#187)

Bumps testfixtures from 6.18.4 to 6.18.5.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: testfixtures dependency-type: indirect update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*3b92776*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3b92776d75ea0e75f5b41bdfb69b78851e0ffc52>`_)

- chore(deps): bump types-setuptools from 57.4.9 to 57.4.10 (#188)

Bumps types-setuptools from 57.4.9 to 57.4.10.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: types-setuptools dependency-type: direct:production update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*dcfaf21*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/dcfaf21f27fd490277de01eb0eb9b59a522d5353>`_)

- chore(deps): bump virtualenv from 20.13.2 to 20.13.3 (#189)

Bumps virtualenv from 20.13.2 to 20.13.3.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: virtualenv dependency-type: indirect update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*e71e5b3*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e71e5b3a46cb6c5c915d9b31eb8e0e815c511a3d>`_)

- chore(deps-dev): bump mypy from 0.931 to 0.940 (#192)

Bumps mypy from 0.931 to 0.940.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``9fce6bf`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9fce6bf853032de9b2eec1f2b20341c8fbe6f639>`_)

- chore: added autopep8 to pre-commit and clarified command in CONTRIBUTING for performance

Signed-off-by: Paul Horton <phorton@sonatype.com> (``5dafb1c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5dafb1c88208caccaf82fc5abea41df0d295d5a4>`_)

- chore: first pass pre-commit config

Signed-off-by: Paul Horton <phorton@sonatype.com> (``fd6ab7a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fd6ab7ab2136c4afd8169fc97e0ee6ecbbef56a7>`_)

- chore: added documentation to CONTRIBUTING guidelines

Signed-off-by: Paul Horton <phorton@sonatype.com> (``67cefe1`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/67cefe1e5f9eb3bdb1d07c29e1ea351937c15bc0>`_)

- chore(deps): bump actions/checkout from 2 to 3 (#184)

Bumps actions/checkout from 2 to 3.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: actions/checkout dependency-type: direct:production update-type: version-update:semver-major …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``a3ed3c7`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a3ed3c712a8a85361a59522efc356ab5194b0999>`_)

- chore(deps): bump actions/setup-python from 2 to 3 (#183)

Bumps actions/setup-python from 2 to 3.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: actions/setup-python dependency-type: direct:production update-type: version-update:semver-major …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``ee79ffa`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ee79ffaaa6155f6890379a847b49a805c1ee7202>`_)

- chore: dependabot prefix `chore`, not eco-system (``c96cea4`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c96cea47f855add5edf2707305ef7b671da7db39>`_)
- chore: make isort and flake8-isort available

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*b211de5*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b211de50b92393e653b9a9f907c66a81b016d870>`_)

- chore: poetry(deps): bump pyparsing from 3.0.6 to 3.0.7 (#140)

Bumps pyparsing from 3.0.6 to 3.0.7.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: pyparsing dependency-type: indirect update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*1bdb798*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1bdb7987a86af967d5a883626346f217a243bfda>`_)

- chore: poetry(deps): bump types-setuptools from 57.4.7 to 57.4.9 (#168)

Bumps types-setuptools from 57.4.7 to 57.4.9.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: types-setuptools dependency-type: direct:production update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*48c3f99*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/48c3f997abf2560b648d85b907c001879e063551>`_)

- chore: poetry(deps): bump filelock from 3.4.0 to 3.4.1 (#116)

Bumps filelock from 3.4.0 to 3.4.1.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: filelock dependency-type: indirect update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*17f1a5f*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/17f1a5f8555675913ea09318848dd28ce96d1c3c>`_)

- chore: poetry(deps): bump attrs from 21.2.0 to 21.4.0 (#113)

Bumps attrs from 21.2.0 to 21.4.0.

- Release notes

- Changelog
- Commits

---

updated-dependencies:

- dependency-name: attrs dependency-type: indirect update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*3c39ae5*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3c39ae5f7435b4e0240e674e47283ac3beb9f2b8>`_)

- chore: poetry(deps): bump typed-ast from 1.5.1 to 1.5.2 (#144)

Bumps typed-ast from 1.5.1 to 1.5.2.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: typed-ast dependency-type: indirect update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*ac5809e*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ac5809e93a3a5c54b04c75bd959216a4b21095ff>`_)

- chore: poetry(deps): bump packageurl-python from 0.9.6 to 0.9.9 (#177)

Bumps packageurl-python from 0.9.6 to 0.9.9.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: packageurl-python dependency-type: direct:production update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*4bfba14*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/4bfba14bfacca773fd2e949e327f94b794fdef0b>`_)

- chore: poetry(deps): bump virtualenv from 20.13.1 to 20.13.2 (#181)

Bumps virtualenv from 20.13.1 to 20.13.2.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: virtualenv dependency-type: indirect update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``20e3368`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/20e3368f35e28187f41ac0652384ea2104d45e35>`_)

Feature

- feat: output errors are verbose

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``bfe8fb1`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bfe8fb18825251fd9f146458122aa06137ec27c0>`_)

Fix

- fix: `version` being optional in JSON output can raise error

Signed-off-by: Paul Horton <phorton@sonatype.com> (``ba0c82f`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ba0c82fbde7ba47502c45caf4fa89e9e4381f482>`_)

Style

- style: sorted all imports

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``4780a84`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/4780a84979d213d6ce6d9527945d532cbd6a8ceb>`_)

Unknown

- 2.1.0

Automatically generated by python-semantic-release (``c58f8f8`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c58f8f8456211fbeac79340b480063791c05f404>`_)

- Merge pull request #198 from CycloneDX/verbose_outout_errors

fix: improved output errors - file/directory is now included (``4618c62`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/4618c62da54f90a67d89583d5339ef0532b7813a>`_)

- updated to be more pythonic

Signed-off-by: Paul Horton <phorton@sonatype.com> (``a1bbf00`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a1bbf001ba9546c998062a0201d4e2562607749e>`_)

- doc: added CONTRIBUTING to public docs doc: included pre-commit hooks in CONTRIBUTING

Signed-off-by: Paul Horton <phorton@sonatype.com> (``f38215f`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f38215f2b370e14f5629edff1ade97734b3a79cd>`_)

- Merge pull request #182 from CycloneDX/sort-imports

style: sort imports (``aa37e56`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/aa37e56964b35642e2bf92f336a767fba1914e2b>`_)

6.1.52 v2.0.0 (2022-02-21)

Breaking

- feat: bump dependencies

BREAKING CHANGE: Adopt PEP-3102

BREAKING CHANGE: Optional Lists are now non-optional Sets

BREAKING CHANGE: Remove concept of DEFAULT schema version - replaced with LATEST schema version

BREAKING CHANGE: Added `BomRef` data type

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*da3f0ca*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/da3f0ca3e8b90b37301c03f889eb089bca649b09>`_)

Chore

- chore: poetry(deps): bump virtualenv from 20.13.0 to 20.13.1 (#167)

Bumps virtualenv from 20.13.0 to 20.13.1.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: virtualenv dependency-type: indirect update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*9e80258*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9e802582bd9b9bdd0e1e91a0af551d3f2190fb5e>`_)

- chore: poetry(deps): bump types-toml from 0.10.3 to 0.10.4 (#166)

Bumps types-toml from 0.10.3 to 0.10.4.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: types-toml dependency-type: direct:production update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*02449f6*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/02449f6102e49f9e2425ab4e5b050f38832e6ba9>`_)

- chore: bump dependencies

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*6c280e7*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6c280e7794466ad9b6f1ce5eb985035bea21eaaa>`_)

Feature

- feat: completed work on #155 (#172)

fix: resolved #169 (part of #155) feat: as part of solving #155, #147 has been implemented

Signed-off-by: Paul Horton <phorton@sonatype.com> (``a926b34`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a926b34c7facb8b3709936fe00b62a0b80338f31>`_)

- feat: support complete model for `bom.metadata` (#162)
- feat: support complete model for `bom.metadata` fix: JSON comparison in unit tests was broken chore: corrected some source license headers

Signed-off-by: Paul Horton <phorton@sonatype.com> (``2938a6c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2938a6c001a5b0b25477241d4ad6601030c55165>`_)

- feat: support for `bom.externalReferences` in JSON and XML #124

Signed-off-by: Paul Horton <phorton@sonatype.com> (``1b733d7`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1b733d75a78e3757010a8049cab5c7d4656dc2a5>`_)

- feat: Complete support for `bom.components` (#155)
- fix: implemented correct `__hash__` methods in models (#153)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``32c0139`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/32c01396251834c69a5b23c82a5554faf8447f61>`_)

- feat: support services in XML BOMs feat: support nested services in JSON and XML BOMs

Signed-off-by: Paul Horton <phorton@sonatype.com> (``9edf6c9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9edf6c940d20a44f5b99c557392a9fa4532b332e>`_)

Fix

- fix: `license_url` not serialised in XML output #179 (#180)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``f014d7c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f014d7c4411de9ed5e9cb877878ae416d85b2d92>`_)

- fix: `Component.bom_ref` is not Optional in our model implementation (in the schema it is) - we generate a UUID if `bom_ref` is not supplied explicitly

Signed-off-by: Paul Horton <phorton@sonatype.com> (``5c954d1`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5c954d1e39ce8509ab36e6de7d521927ad3c997c>`_)

- fix: temporary fix for `__hash__` of Component with `properties` #153

Signed-off-by: Paul Horton <phorton@sonatype.com> (``a51766d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a51766d202c3774003dd7cd8c115b2d9b3da1f50>`_)

- fix: further fix for #150

Signed-off-by: Paul Horton <phorton@sonatype.com> (``1f55f3e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1f55f3edfeacfc515ef0b5e493c27dd6e14861d6>`_)

- fix: regression introduced by first fix for #150

Signed-off-by: Paul Horton <phorton@sonatype.com> (``c09e396`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c09e396b98c484d1d3d509a5c41746133fe41276>`_)

- fix: Components with no version (optional since 1.4) produce invalid BOM output in XML #150

Signed-off-by: Paul Horton <phorton@sonatype.com> (``70d25c8`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/70d25c8c162e05a5992761ccddbad617558346d1>`_)

- fix: `expression` not supported in Component Licsnes for version 1.0

Signed-off-by: Paul Horton <phorton@sonatype.com> (``15b081b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/15b081bd1891566dbe00e18a8b21d3be87154f72>`_)

Test

- test: refactor to work on PY < 3.10

Signed-off-by: Paul Horton <phorton@sonatype.com> (``0ce5de6`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0ce5de6a223e10161a8b864d0115e95d849d5e87>`_)

- test: refactored fixtures for tests which has uncovered #150, #151 and #152

Signed-off-by: Paul Horton <phorton@sonatype.com> (``df43a9b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/df43a9bff4b8360234bf50058ded82e44e2df082>`_)

Unknown

- 2.0.0

Automatically generated by python-semantic-release (``a4af3dc`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a4af3dccbddf4ea91b277746d2305fadf6078ed8>`_)

- Merge pull request #148 from CycloneDX/feat/add-bom-services (``631e400`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/631e4009340f4466fb45f25bbf3ce7ffa4d8adca>`_)
- Merge branch 'main' into feat/add-bom-services (``9a32351`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9a3235155bd04450c6e520ee6de04b2d6f2c5d0a>`_)
- doc: added RTD badge to README

Signed-off-by: Paul Horton <phorton@sonatype.com> (``b20d9d1`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b20d9d1aceebfa8bae21250e6ae39234caffbb0e>`_)

- implemented `__str__` for `BomRef`

Signed-off-by: Paul Horton <phorton@sonatype.com> (``670bde4`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/670bde47a8a60db764aa706797f1d8ed7cf2c227>`_)

- Continuation of #170 - missed updating Vulnerability to use `BomRef` (#175)
- BREAKING CHANGE: added new model `BomRef` unlocking logic later to ensure uniquness and dependency references

Signed-off-by: Paul Horton <phorton@sonatype.com>

- updated Vulnerability to also use new `BomRef` model

Signed-off-by: Paul Horton <phorton@sonatype.com> (``0d82c01`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0d82c019afce3e4aefe56bff9607cfd60186c6b0>`_)

- BREAKING CHANGE: added new model `BomRef` unlocking logic later to ensure uniquness and dependency references (#174)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``d189f2c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d189f2c16870deb683e62cd06a6072b008eab05d>`_)

- BREAKING CHANGE: replaced concept of default schema version with latest supported #171 (#173)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*020fcf0*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/020fcf03ef3985dac82a38b8810d6d6cd301809c>`_)

- BREAKING CHANGE: Updated default schema version to 1.4 from 1.3 (#164)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*9b6ce4b*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9b6ce4bd7b5a2a332e9f01f93db57b78f65af048>`_)

- BREAKING CHANGE: update models to use `Set` rather than `List` (#160)
- BREAKING CHANGE: update models to use `Set` and `Iterable` rather than `List[..]` BREAKING CHANGE: update final models to use `@property` wip

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*142b8bf*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/142b8bf4dbb2e61d131b7ca2ec332aac472ef3cd>`_)

- removed unnecessary calls to `hash()` in `__hash__()` methods as pointed out by @jkowalleck

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*0f1fd6d*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0f1fd6dfdd41073cbdbb456cf019c7f2ed9e2175>`_)

- BREAKING CHANGE: adopted PEP-3102 for model classes (#158)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*b3c8d9a*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b3c8d9a676190f20dfc4ab1b915c1e53c4ac5a82>`_)

- doc: added page to docs to call out which parts of the specification this library supports

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*41a4be0*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/41a4be0cedcd26b6645b6e3606cce8e3708c569f>`_)

- attempt to resolve Lift finding

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*2090c08*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2090c0868ca82c4b53c6ffc6f439c0d675147601>`_)

- removed unused imports

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*a35d540*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a35d540c97b898eb152f453003f46ce0e18b7ea6>`_)

- WIP on `bom.services`
- WIP but a lil hand up for @madpah

Signed-off-by: Jeffry Hesse <5544326+DarthHater@users.noreply.github.com>

- chore: added missing license header

Signed-off-by: Paul Horton <phorton@sonatype.com>

- No default values for required fields
- Add Services to BOM
- Typo fix
- aligned classes with standards, commented out Signature work for now, added first tests for Services

Signed-off-by: Paul Horton <phorton@sonatype.com>

- addressed standards

Signed-off-by: Paul Horton <phorton@sonatype.com>

- 1.2.0

Automatically generated by python-semantic-release

Signed-off-by: Paul Horton <phorton@sonatype.com>

- feat: `bom-ref` for Component and Vulnerability default to a UUID (#142)
- feat: `bom-ref` for Component and Vulnerability default to a UUID if not supplied ensuring they have a unique value #141

Signed-off-by: Paul Horton <phorton@sonatype.com>

- doc: updated documentation to reflect change

Signed-off-by: Paul Horton <phorton@sonatype.com>

- patched other tests to support UUID for bom-ref

Signed-off-by: Paul Horton <phorton@sonatype.com>

- better syntax

Signed-off-by: Paul Horton <phorton@sonatype.com>

- 1.3.0

Automatically generated by python-semantic-release

Signed-off-by: Paul Horton <phorton@sonatype.com>

- WIP but a lil hand up for @madpah

Signed-off-by: Jeffry Hesse <5544326+DarthHater@users.noreply.github.com> Signed-off-by: Paul Horton <phorton@sonatype.com>

- chore: added missing license header

Signed-off-by: Paul Horton <phorton@sonatype.com>

- aligned classes with standards, commented out Signature work for now, added first tests for Services

Signed-off-by: Paul Horton <phorton@sonatype.com>

- removed signature from this branch

Signed-off-by: Paul Horton <phorton@sonatype.com>

- Add Services to BOM
- Typo fix
- addressed standards

Signed-off-by: Paul Horton <phorton@sonatype.com>

- resolved typing issues from merge

Signed-off-by: Paul Horton <phorton@sonatype.com>

- added a bunch more tests for JSON output

Signed-off-by: Paul Horton <phorton@sonatype.com>

Co-authored-by: Paul Horton <phorton@sonatype.com> Co-authored-by: github-actions <action@github.com> (``*b45ff18*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b45ff187056893c5fb294cbf9de854fd130bb7be>`_)

6.1.53 v1.3.0 (2022-01-24)

Feature

- feat: `bom-ref` for Component and Vulnerability default to a UUID (#142)
- feat: `bom-ref` for Component and Vulnerability default to a UUID if not supplied ensuring they have a unique value #141

Signed-off-by: Paul Horton <phorton@sonatype.com>

- doc: updated documentation to reflect change

Signed-off-by: Paul Horton <phorton@sonatype.com>

- patched other tests to support UUID for bom-ref

Signed-off-by: Paul Horton <phorton@sonatype.com>

- better syntax

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*3953bb6*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3953bb676f423c325ca4d80f3fcee33ad042ad93>`_)

Unknown

- 1.3.0

Automatically generated by python-semantic-release (``*4178181*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/41781819e2de8f650271e7de11d395fa43939f22>`_)

6.1.54 v1.2.0 (2022-01-24)

Feature

- feat: add CPE to component (#138)
- Added CPE to component

Setting CPE was missing for component, now it is possible to set CPE and output CPE for a component.

Signed-off-by: Jens Lucius <jens.lucius@de.bosch.com>

- Fixing problems with CPE addition
- Fixed styling errors
- Added reference to CPE Spec
- Adding CPE parameter as last parameter to not break arguments

Signed-off-by: Jens Lucius <jens.lucius@de.bosch.com>

- Again fixes for Style and CPE reference

Missing in the last commit

Signed-off-by: Jens Lucius <jens.lucius@de.bosch.com>

- Added CPE as argument before deprecated arguments

Signed-off-by: Jens Lucius <jens.lucius@de.bosch.com>

- Added testing for CPE addition and error fixing
- Added output tests for CPE in XML and JSON
- Fixes style error in components
- Fixes order for CPE output in XML (CPE has to come before PURL)

Signed-off-by: Jens Lucius <jens.lucius@de.bosch.com>

- Fixed output tests

CPE was still in the wrong position in one of the tests - fixed

Signed-off-by: Jens Lucius <jens.lucius@de.bosch.com>

- Fixed minor test fixtures issues
- cpe was still in wrong position in 1.2 JSON
- Indentation fixed in 1.4 JSON

Signed-off-by: Jens Lucius <jens.lucius@de.bosch.com>

- Fixed missing comma in JSON 1.2 test file

Signed-off-by: Jens Lucius <jens.lucius@de.bosch.com> (``269ee15`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/269ee155f203d5771c56edb92f7279466bf2012f>`_)

Unknown

- 1.2.0

Automatically generated by python-semantic-release (``97c215c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/97c215cf0c4e8c315ed84cbcb92b22c6b7bcd8c2>`_)

6.1.55 v1.1.1 (2022-01-19)

Fix

- fix: bump dependencies (#136)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``18ec498`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/18ec4987f6aa4a259d30000a19aa6ee1d49681d1>`_)

Unknown

- 1.1.1

Automatically generated by python-semantic-release (``dec63de`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/dec63de950e0ad81cbb51373b0e647bce551297e>`_)

6.1.56 v1.1.0 (2022-01-13)

Feature

- feat: add support for `bom.metadata.component` (#118)
- Add support for metadata component

Part of #6

Signed-off-by: Artem Smotrakov <asmotrakov@riotgames.com>

- Better docs and simpler ifs

Signed-off-by: Artem Smotrakov <asmotrakov@riotgames.com> (``1ac31f4`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1ac31f4cb14b6c466e092ff38ee2aa472c883c5d>`_)

Unknown

- 1.1.0

Automatically generated by python-semantic-release (``d4007bd`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d4007bd5986173eb2645eebcdd2c6405150f1456>`_)

6.1.57 v1.0.0 (2022-01-13)

Chore

- chore: attempt to produce manual GitHub action to release a RC version

Signed-off-by: Paul Horton <phorton@sonatype.com> (``3058afc`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3058afc42fa632be6a0efb9ef509612d8833e07b>`_)

- chore: attempt to produce manual GitHub action to release a RC version

Signed-off-by: Paul Horton <phorton@sonatype.com> (``6799e63`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6799e632d2eb1d3cee0042c2350477a74bcdce83>`_)

- chore: disable poetry-cache in gh-workflow (#112)

closes #91

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``42f7952`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/42f7952fad512c84fd0a4d08c564af43d8bc5c87>`_)

- chore: removed pdoc3 from main dev dependencies as now covered in docs/requirements.txt

Signed-off-by: Paul Horton <phorton@sonatype.com> (``89d8382`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/89d8382dc0e8bedb129ca0bbbd95922ea104f95c>`_)

- chore: isolate dependencies for building documentation (#107)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``f2403f6`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f2403f66c21e55de552b10c473332a1ea72b25bf>`_)

- chore: bump `flake8` to v4 and add `autopep8` (#93)
- chore: bump `flake8` to v4 and add `autopep8`

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- chore: make pep8 known in the contrib docs

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*6553dbf*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6553dbfefcf6865b28b72771a9a08f1387dbdf11>`_)

- chore: poetry(deps-dev): bump mypy from 0.910 to 0.920 (#103)

Bumps mypy from 0.910 to 0.920.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: mypy dependency-type: direct:development update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com>

Co-authored-by: dependabot[bot] <49699333+dependabot[bot]@users.noreply.github.com> (``*fdd20ca*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fdd20ca4be71be78b578f756f46b44d829a76212>`_)

Unknown

- Manually generated release (``*3509fb6*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3509fb643af12cc4393309a006c6bbe63b1bd674>`_)
- Support for CycloneDX schema version 1.4 (#108)

BREAKING CHANGE: Support for CycloneDX 1.4. This includes:

- Support for `tools` having `externalReferences`
- Allowing `version` for a `Component` to be optional in 1.4
- Support for `releaseNotes` per `Component`
- Support for the core schema implementation of Vulnerabilities (VEX)

Other changes included in this PR:

- Unit tests now include schema validation (we've left schema validation out of the core library due to dependency bloat)
- Fixes to ensure schema is adhered to in 1.0
- URI's are now used throughout the library through a new `XsUri` class to provide URI validation
- Documentation is now hosted on readthedocs.org (https://cyclonedx-python-library.readthedocs.io/)
- `$schema` is now included in JSON BOMs
- Concrete Parsers how now been moved into downstream projects to keep this libraries focus on modelling and outputting CycloneDX - see https://github.com/CycloneDX/cyclonedx-python
- Added reference to release of this library on Anaconda

Signed-off-by: Paul Horton <phorton@sonatype.com>

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

Co-authored-by: Paul Horton <phorton@sonatype.com>

Co-authored-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*7fb6da9*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7fb6da9166050333ae5db7e35ab792b9bdee48d4>`_)

- Merge branch 'main' of github.com:CycloneDX/cyclonedx-python-lib (``d26970b` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d26970bcc52568645c303f060d71cbc25edbfe78>`_)
- Update CONTRIBUTING.md (``4448d9b` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/4448d9b4846a7dfb9eeee355d41fbb100a48d388>`_)

6.1.58 v0.12.3 (2021-12-15)

Fix

- fix: removed requirements-parser as dependency (temp) as not available for Python 3 as Wheel (#98)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``3677d9f` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3677d9fd584b7c0eb715954bb7b8adc59c0bc9b1>`_)

Unknown

- 0.12.3

Automatically generated by python-semantic-release (``cfc9d38` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/cfc9d382aea3f69f79d50a4fbb8607346f86ce03>`_)

6.1.59 v0.12.2 (2021-12-09)

Fix

- fix: tightened dependency `packageurl-python` (#95)

fixes #94

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``eb4ae5c` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/eb4ae5ca8842877b780a755b6611feef847bdb8c>`_)

Unknown

- 0.12.2

Automatically generated by python-semantic-release (``54b9f74` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/54b9f744be28b53795bd03e78576eed15b70c10a>`_)

6.1.60 v0.12.1 (2021-12-09)

Chore

- chore: reordered deps & updated poetry lock

Merge pull request #90 from CycloneDX/update-poetry-lock (``d8c7ee2` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d8c7ee2915c23d22bc49c9d562a052783ea7ea87>`_)

- chore: updated poetry lock

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``91b97be` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/91b97bedfa0a22598e9f4e8731bcf7293bc7d57d>`_)

Fix

- fix: further loosened dependency definitions

see #44

updated some locked dependencies to latest versions

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*8bef6ec*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8bef6ecad36f51a003b266d776c9520d33e06034>`_)

Unknown

- 0.12.1

Automatically generated by python-semantic-release (``*43fc36e*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/43fc36ebc966ac511e5b7dbff9b0bef6f88d5d2c>`_)

6.1.61 v0.12.0 (2021-12-09)

Ci

- ci: update to run tox for both our favoured versions of dependencies and lowest supported versions
- add tox env for minimal required dependencies

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- try to fix `TypedDict` typing

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- fix: typing definitions to be PY 3.6 compatible

Signed-off-by: Paul Horton <phorton@sonatype.com>

- fix: typing definitions to be PY 3.6 compatible

Signed-off-by: Paul Horton <phorton@sonatype.com>

- straigtened up `sys.version_info` constraints/code-branches

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- removed unused type ignores

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- try to fix type variants

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- try to fix type variants

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- typing for py3.6

Signed-off-by: Paul Horton <phorton@sonatype.com>

- fixed invalid unittest

Signed-off-by: Paul Horton <phorton@sonatype.com>

- typing for py3.6

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- mypy silence `warn_unused_ignores`

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- mypy in tox for lowest version is pinned

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

Co-authored-by: Paul Horton <phorton@sonatype.com> (``07ebedc`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/07ebedcbab1554970496780bb8bf167f6fe4ad5c>`_)

Feature

- feat: loosed dependency versions to make this library more consumable
- feat: lowering minimum dependency versions

Signed-off-by: Paul Horton <phorton@sonatype.com>

- feat: lowering minimum dependency versions

Signed-off-by: Paul Horton <phorton@sonatype.com>

- feat: lowering minimum dependency versions - importlib-metadata raising minimum to ensure we get a typed library

Signed-off-by: Paul Horton <phorton@sonatype.com>

- feat: lowering minimum dependency versions - importlib-metadata raising minimum to ensure we get a typed library

Signed-off-by: Paul Horton <phorton@sonatype.com>

- feat: lowering minimum version for importlib-metadata to 3.4.0 with modified import statement

Signed-off-by: Paul Horton <phorton@sonatype.com> (``55f10fb`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/55f10fb5524dafa68112c0836806c27bdd74fcbe>`_)

Unknown

- 0.12.0

Automatically generated by python-semantic-release (``1a907ea`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1a907eae0a3436844ffc2782b990c4b502f409e6>`_)

- Merge pull request #88 from CycloneDX/contributing-file

initial CONTRIBUTING file (``20035bb`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/20035bb5dde8dd3b619b200aec7037c338b18c74>`_)

- initial CONTRIBUTING file

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``6ffe14d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6ffe14d4d51d246cda66ce99ee20893ede8d017f>`_)

- CHORE: poetry(deps): bump filelock from 3.3.2 to 3.4.0

poetry(deps): bump filelock from 3.3.2 to 3.4.0 (``e144aa2`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e144aa29a0fd61483f4940da08ff542c9c3c3332>`_)

- CHORE: poetry(deps): bump types-setuptools from 57.4.2 to 57.4.4

poetry(deps): bump types-setuptools from 57.4.2 to 57.4.4 (``5fcdcb7` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5fcdcb701a9da5c9a786e0fe690bfd0a8d5d4e0c>`_)

- poetry(deps): bump filelock from 3.3.2 to 3.4.0

Bumps filelock from 3.3.2 to 3.4.0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: filelock dependency-type: indirect update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com> (``8d4520e` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8d4520ee3ee781a3a2f4db879e79e38b40fe4829>`_)

- CHORE: poetry(deps-dev): bump flake8-bugbear from 21.9.2 to 21.11.29

poetry(deps-dev): bump flake8-bugbear from 21.9.2 to 21.11.29 (``fc6e3ac` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fc6e3acd5a1875a27e3b8037ad3b9a794598c894>`_)

- poetry(deps): bump types-setuptools from 57.4.2 to 57.4.4

Bumps types-setuptools from 57.4.2 to 57.4.4.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: types-setuptools dependency-type: direct:production update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com> (``00dcbb8` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/00dcbb80d25c00b2b9bd4f6b765275cd956b33fa>`_)

- CHORE: poetry(deps): bump importlib-metadata from 4.8.1 to 4.8.2

poetry(deps): bump importlib-metadata from 4.8.1 to 4.8.2 (``28f9676` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/28f96769e653c3b7c76cb07ba1a4ecbbc43ab46c>`_)

- poetry(deps-dev): bump flake8-bugbear from 21.9.2 to 21.11.29

Bumps flake8-bugbear from 21.9.2 to 21.11.29.

- Release notes
- Commits

---

updated-dependencies:

- dependency-name: flake8-bugbear dependency-type: direct:development update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com> (``1eec2e8` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1eec2e8aab5f31f3070be34eccfd8791ef2edcca>`_)

- CHORE: poetry(deps-dev): bump coverage from 6.1.2 to 6.2

poetry(deps-dev): bump coverage from 6.1.2 to 6.2 (``bdd9365` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bdd93650a64ce2385f4f29bc1f20df6530e9012c>`_)

- CHORE: poetry(deps): bump mako from 1.1.5 to 1.1.6

poetry(deps): bump mako from 1.1.5 to 1.1.6 (``33d3ecc` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/33d3ecc80f47c947d2fc2b13743471dd6dc941ab>`_)

- poetry(deps-dev): bump coverage from 6.1.2 to 6.2

Bumps coverage from 6.1.2 to 6.2.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: coverage dependency-type: direct:development update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com> (``be1af9b` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/be1af9b9955a31b6c1a8627010bfd4d932c9f9f1>`_)

- DOCS: fix README shields & links (``43b1121` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/43b112128acd9e28a47e46d8691ead46e39b288e>`_)
- doc: readme maintenance - shields & links (#72)
- README: restructure links

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- README: add lan to fenced code blocks

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- README: fix some formatting

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- README: modernized shields

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- README: harmonize links

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- README: add language to code fences

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- README: markdown fixes

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- README: removed py version shield

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``3d0ea2f` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3d0ea2f4c6ee5c2dedf1abb779f46543896fff4a>`_)

- poetry(deps): bump mako from 1.1.5 to 1.1.6

Bumps mako from 1.1.5 to 1.1.6.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: mako dependency-type: indirect update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com> (``3344b86`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3344b862490ecb419c9b1f74bd7548ddcf392329>`_)

- Merge pull request #47 from CycloneDX/dependabot/pip/filelock-3.3.2

poetry(deps): bump filelock from 3.3.1 to 3.3.2 (``3f967b3`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3f967b3d0ec47ba5bcc1cdd8fb29970ba69d7aed>`_)

- FIX: update Conda package parsing to handle `build` containing underscore (#66)
- fix: update conda package parsing to handle `build` containing underscore

Signed-off-by: Paul Horton <phorton@sonatype.com>

- updated some typings

Signed-off-by: Paul Horton <phorton@sonatype.com> (``2c6020a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2c6020a208aa1c0fd13ab337db6343ad1d2d5c43>`_)

- poetry(deps): bump importlib-metadata from 4.8.1 to 4.8.2

Bumps importlib-metadata from 4.8.1 to 4.8.2.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: importlib-metadata dependency-type: direct:production update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com> (``003f6b4`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/003f6b410e0e32e8c454ad157999b031471baf6f>`_)

- poetry(deps): bump filelock from 3.3.1 to 3.3.2

Bumps filelock from 3.3.1 to 3.3.2.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: filelock dependency-type: indirect update-type: version-update:semver-patch …

Signed-off-by: dependabot[bot] <support@github.com> (``55022b7`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/55022b7a63763436d193cefda6d6a4e0ad36fb40>`_)

- Merge pull request #45 from CycloneDX/dependabot/pip/importlib-resources-5.4.0

poetry(deps): bump importlib-resources from 5.3.0 to 5.4.0 (``*b8acf9f*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b8acf9f3e087f37c2f9afded2d8555c053f09a43>`_)

- Merge pull request #70 from CycloneDX/dependabot/pip/pyparsing-3.0.6

poetry(deps): bump pyparsing from 3.0.5 to 3.0.6 (``*faa8628*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/faa862813e27bb4b828f6116c95961b156cd7547>`_)

- Merge pull request #69 from CycloneDX/dependabot/pip/coverage-6.1.2

poetry(deps-dev): bump coverage from 6.1.1 to 6.1.2 (``*eba56dc*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/eba56dc6512304e2956563d173bdb363b785fa50>`_)

- poetry(deps): bump pyparsing from 3.0.5 to 3.0.6

Bumps pyparsing from 3.0.5 to 3.0.6.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: pyparsing dependency-type: indirect update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com> (``*4f2b2d8*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/4f2b2d89291b1c20385ce6431959586acfeab1cd>`_)

- poetry(deps-dev): bump coverage from 6.1.1 to 6.1.2

Bumps coverage from 6.1.1 to 6.1.2.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: coverage dependency-type: direct:development update-type: version-update:semver-patch ...

Signed-off-by: dependabot[bot] <support@github.com> (``*1d0f5ea*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1d0f5ea2ed5dfb38ce1d1d8170773cb880f228dc>`_)

6.1.62 v0.11.1 (2021-11-10)

Fix

- fix: constructor for `Vulnerability` to correctly define `ratings` as optional

Signed-off-by: William Woodruff <william@trailofbits.com> (``*395a0ec*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/395a0ec14ebcba8e0849a0ced30ec4163c42fa7a>`_)

Unknown

- 0.11.1

Automatically generated by python-semantic-release (``a80f87a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a80f87a588f8b52bfd8e9c5b12edf0fdde56c510>`_)

- FEAT: Support Python 3.10 (#64)
- fix: tested with Python 3.10

Signed-off-by: Paul Horton <phorton@sonatype.com>

- added trove classifier for Python 3.10

Signed-off-by: Paul Horton <phorton@sonatype.com>

- fix: upgrade Poetry version to workaround issue between Poetry and Python 3.10 (see: https://github.com/python-poetry/poetry/issues/4210)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``385b835`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/385b835f44fadb0f227b6a8ac992b0c73afc6ef0>`_)

- poetry(deps): bump importlib-resources from 5.3.0 to 5.4.0

Bumps importlib-resources from 5.3.0 to 5.4.0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: importlib-resources dependency-type: indirect update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com> (``a1dd775`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a1dd7752459b70b432784ec2b7d8a1cb24a916a9>`_)

6.1.63 v0.11.0 (2021-11-10)

Feature

- feat: Typing & PEP 561
- adde file for type checkers according to PEP 561

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- added static code analysis as a dev-test

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- added the "typed" trove

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- added `flake8-annotations` to the tests

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- added type hints

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- further typing updates

Signed-off-by: Paul Horton <phorton@sonatype.com>

- further typing additions and test updates

Signed-off-by: Paul Horton <phorton@sonatype.com>

- further typing

Signed-off-by: Paul Horton <phorton@sonatype.com>

- further typing - added type stubs for toml and setuptools

Signed-off-by: Paul Horton <phorton@sonatype.com>

- further typing

Signed-off-by: Paul Horton <phorton@sonatype.com>

- typing work

Signed-off-by: Paul Horton <phorton@sonatype.com>

- coding standards

Signed-off-by: Paul Horton <phorton@sonatype.com>

- fixed tox and mypy running in correct python version

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- supressed mypy for `cyclonedx.utils.conda.parse_conda_json_to_conda_package`

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- fixed type hints

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- fixed some typing related flaws

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

- added flake8-bugbear for code analysis

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com>

Co-authored-by: Paul Horton <phorton@sonatype.com> (``*9144765*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/91447656c0914ceb2af2e4b7282292ec7b93f5bf>`_)

Unknown

- 0.11.0

Automatically generated by python-semantic-release (``*7262783*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7262783dbcf5823065670f3f7cbba0ce25b3a4ea>`_)

- Merge pull request #41 from jkowalleck/improv-abstract

fixed some abstract definitions (``*f34e2c2*`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f34e2c2bc7aed20968a5ac69337ed484d097af3b>`_)

- Merge pull request #42 from jkowalleck/improv-pipenv

slacked pipenv parser (``08bc4ab`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/08bc4ab2b01c76d7472a558cae02deab0485c61c>`_)

- Merge pull request #43 from jkowalleck/improv-conda-typehints

fixed typehints/docs in `_BaseCondaParser` (``931016d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/931016d9b700280692903db5aa653d390a80bd63>`_)

- Merge pull request #54 from jkowalleck/create-CODEOWNERS

created CODEOWNERS (``7f28bef`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7f28bef15ed0b9ed6af88286d5f6dcc0726b6feb>`_)

- Merge pull request #56 from CycloneDX/dependabot/pip/py-1.11.0

poetry(deps): bump py from 1.10.0 to 1.11.0 (``f1cda3c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f1cda3c3ba859336d70da36d4966bc7c247af97a>`_)

- Merge pull request #58 from CycloneDX/dependabot/pip/pyparsing-3.0.5

poetry(deps): bump pyparsing from 2.4.7 to 3.0.5 (``0525439`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0525439d2237684ce531449d19e60456fc46d26b>`_)

- Merge pull request #19 from CycloneDX/dependabot/pip/zipp-3.6.0

poetry(deps): bump zipp from 3.5.0 to 3.6.0 (``c54c968`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c54c96853e3325571dee26038e965279d5b9cfe2>`_)

- poetry(deps): bump py from 1.10.0 to 1.11.0

Bumps py from 1.10.0 to 1.11.0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: py dependency-type: indirect update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com> (``330711f`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/330711fe911739ac9119a0721f7f7bde6e1389e4>`_)

- Merge pull request #57 from CycloneDX/dependabot/pip/coverage-6.1.1

poetry(deps-dev): bump coverage from 5.5 to 6.1.1 (``fa55e5c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fa55e5ceef65749ccbf6bd0303db649346c79019>`_)

- poetry(deps): bump pyparsing from 2.4.7 to 3.0.5

Bumps pyparsing from 2.4.7 to 3.0.5.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: pyparsing dependency-type: indirect update-type: version-update:semver-major ...

Signed-off-by: dependabot[bot] <support@github.com> (``*3bedaff*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3bedaffc7f52026348cc6e2a38ba193ba71d4f29>`_)

- Merge pull request #55 from CycloneDX/dependabot/pip/virtualenv-20.10.0

poetry(deps): bump virtualenv from 20.8.1 to 20.10.0 (``*4c3df85*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/4c3df857eba656f1ccb51ba9ad6af2cb49226747>`_)

- CI/CT runs on main & master branch (``*2d0df7b*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2d0df7bacf4ead54eee7378ede8626cc93fce3df>`_)
- poetry(deps-dev): bump coverage from 5.5 to 6.1.1

Bumps coverage from 5.5 to 6.1.1.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: coverage dependency-type: direct:development update-type: version-update:semver-major …

Signed-off-by: dependabot[bot] <support@github.com> (``*e322d74*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e322d7476b4a17b012d27c26683809bd1dee86b1>`_)

- poetry(deps): bump virtualenv from 20.8.1 to 20.10.0

Bumps virtualenv from 20.8.1 to 20.10.0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: virtualenv dependency-type: indirect update-type: version-update:semver-minor …

Signed-off-by: dependabot[bot] <support@github.com> (``*3927cdc*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3927cdcd2c37af23543832dbfae2d087cb09787c>`_)

- created CODEOWNERS

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*e8e499c*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e8e499cb2b74f9d7e7afe4d0f00e1725eabb655e>`_)

- fixed typehints/docs in `_BaseCondaParser`

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*af6ddfd*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/af6ddfdc8c7cbdd1bade5ea0c89896ca9791eb3d>`_)

- slacked pipenv parser

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*a3572ba*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a3572ba61ca537de8efd0855c774819a963cd212>`_)

- fixed some abstract definitions

Signed-off-by: Jan Kowalleck <jan.kowalleck@gmail.com> (``*9e67998*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9e67998e53558363b2c76c75f13bb2772fb5a22d>`_)

6.1.64 v0.10.2 (2021-10-21)

Fix

- fix: correct way to write utf-8 encoded files

Signed-off-by: Paul Horton <phorton@sonatype.com> (``49f9369`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/49f9369b3eba47a3a8d1bcc505546d7dfaf4c5fe>`_)

Unknown

- 0.10.2

Automatically generated by python-semantic-release (``79538e9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/79538e92834e548a3f9697388a47efa3b27da678>`_)

6.1.65 v0.10.1 (2021-10-21)

Ci

- ci: disable git automatic line ending conversions

Signed-off-by: Paul Horton <phorton@sonatype.com> (``350c097`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/350c097d1dcad367913f65d1026288777e5e4ba4>`_)

- ci: update to run on OSX and Windows

Signed-off-by: Paul Horton <phorton@sonatype.com> (``6588c4c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6588c4cc37351ac006eded165284f793f9f98bc2>`_)

Fix

- fix: ensure output to file is UTF-8

Signed-off-by: Paul Horton <phorton@sonatype.com> (``a10da20`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a10da20865e90e9a0a5bb1e12fba9cfd23970c39>`_)

- fix: ensure output to file is UTF-8

Signed-off-by: Paul Horton <phorton@sonatype.com> (``193bf64`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/193bf64cdb19bf6fb9662367402dcf7eaab8dd1a>`_)

Unknown

- 0.10.1

Automatically generated by python-semantic-release (``e6451a3`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e6451a39ee18fcf49287a8f685df730846e965b7>`_)

- Merge pull request #40 from CycloneDX/fix/issue-39-windows-UnicodeEncodeError

FIX: Resolve file encoding issues on Windows (``48329e0`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/48329e033e499f4b9a2c204b2fe5c7c512689605>`_)

- remove memoryview from sha1 file hashing

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*a56be0f*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a56be0f2044c1c867c383a7ed26f5fce4097d21a>`_)

- added debug to CI to aid understanding of miss matching SHA1 hashes on Windows

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*10c6b51*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/10c6b51ec1fb8fc816002fda96e551ff0e430941>`_)

6.1.66 v0.10.0 (2021-10-20)

Feature

- feat: add support for Conda

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*bd29c78*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bd29c782d39a4956f482b9e4de20d7f829beefba>`_)

Unknown

- 0.10.0

Automatically generated by python-semantic-release (``*eea3598*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/eea35980ab121899d46178ec10e90058d0e1be45>`_)

- Merge pull request #38 from CycloneDX/feat/conda-support

feat: add support for Conda (``*ee5d36d*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ee5d36dd677abfb1ba5600b44abf45cb2612b792>`_)

- add support pre Python 3.8

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*2d01116*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2d011165e36d03c8d82c7b92b56f1aeec9c18cd6>`_)

- doc: updated documentation with Conda support (and missed updates for externalReferences)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*57e9dc7*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/57e9dc7b2adcfa2bac60a854c91bf77947e8e9cf>`_)

6.1.67 v0.9.1 (2021-10-19)

Fix

- fix: missing check for Classifiers in Environment Parser

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*b7fa38e*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b7fa38e9740bbc5b4c406410df37c3b34818010c>`_)

Unknown

- 0.9.1

Automatically generated by python-semantic-release (``f132c92`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f132c92bf38f1c173b381f18817f0f86b6ddde85>`_)

- Merge branch 'main' of github.com:CycloneDX/cyclonedx-python-lib (``51a1e50`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/51a1e50aad27c1f862812031be74281e839815df>`_)

6.1.68 v0.9.0 (2021-10-19)

Feature

- feat: add support for parsing package licenses when using the `Environment` Parsers

Signed-off-by: Paul Horton <phorton@sonatype.com> (``c414eaf`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c414eafde2abaca1005a2a0af6993fcdc17897d3>`_)

Unknown

- 0.9.0

Automatically generated by python-semantic-release (``ad65564`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ad6556462d92381dcd8494ca93496ea796282565>`_)

- Merge pull request #36 from CycloneDX/feat/add-license-support

Add support for parsing package licenses from installed packages (``d45f75b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d45f75b88611ab97f39bde672cbdd9e8ff71dd3e>`_)

6.1.69 v0.8.3 (2021-10-14)

Fix

- fix: coding standards violations

Signed-off-by: Paul Horton <phorton@sonatype.com> (``00cd1ca`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/00cd1ca20899b6861b1b959611a3556ffad36832>`_)

- fix: handle `Pipfile.lock` dependencies without an `index` specified fix: multiple fixes in variable scoping to prevent accidental data sharing

Signed-off-by: Paul Horton <phorton@sonatype.com> (``26c62fb`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/26c62fb996c4b1b2bf719e10c9072cf4fbadab9f>`_)

Unknown

- 0.8.3

Automatically generated by python-semantic-release (``91f9a8b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/91f9a8bb60fe8faddd86268c0ede89cd0caa5a76>`_)

- Merge pull request #34 from CycloneDX/fix/issue-33-pipfile-lock-parse-failure

BUG: Fixe for `Pipfile.lock` parsing + accidental data sharing issues identified during testing (``4079323`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/4079323617263886319ddcf80ee1d77909a40b69>`_)

6.1.70 v0.8.2 (2021-10-14)

Fix

- fix: add namespace and subpath support to Component to complete PackageURL Spec support

Signed-off-by: Paul Horton <phorton@sonatype.com> (``780adeb`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/780adebe3861ef08eb1e8817a5e9e3451c0a2137>`_)

Unknown

- 0.8.2

Automatically generated by python-semantic-release (``298318f`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/298318fdbf252115f874eb544c2d1f24abb6ab5a>`_)

- Merge pull request #32 from CycloneDX/feat/full-packageurl-support

Add `namespace` and `subpath` support to `Component` (``bb3af91`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bb3af916f1ff0e224d9c197596570bca98ea4525>`_)

6.1.71 v0.8.1 (2021-10-12)

Fix

- fix: multiple hashes being created for an externalRefernce which is not as required

Signed-off-by: Paul Horton <phorton@sonatype.com> (``970d192`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/970d19202d13d4becbbf040b3a9fb115dd7a0795>`_)

Unknown

- 0.8.1

Automatically generated by python-semantic-release (``70689a2`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/70689a21edfd5f17cd2aabc09d4579646a4f1633>`_)

6.1.72 v0.8.0 (2021-10-12)

Feature

- feat: add support for `externalReferneces` for `Components` and associated enhancements to parsers to obtain information where possible/known

Signed-off-by: Paul Horton <phorton@sonatype.com> (``a152852`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a152852b361bbb7a69c9f7ab61ae7ea6dcffd214>`_)

Unknown

- 0.8.0

Automatically generated by python-semantic-release (``7a49f9d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7a49f9d8cd791e9b1a7e1a8587e589e3b8319ec7>`_)

- Merge pull request #29 from CycloneDX/feat/component-external-references

FEATURE: Add support for `externalReferences` against `Components` (``bdee0ea`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bdee0ea277d9f378b3a5e225c2ac3d8e20e2c53c>`_)

- doc: notable improvements to API documentation generation (added search, branding, a little styling)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``e7a5b5a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e7a5b5a2c5b5681a75a24e9739d13ead01f362e3>`_)

6.1.73 v0.7.0 (2021-10-11)

Feature

- feat: support for pipenv.lock file parsing

Signed-off-by: Paul Horton <phorton@sonatype.com> (``68a2dff`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/68a2dffc770d40f693b6891a580d1f7d8018f71c>`_)

Unknown

- 0.7.0

Automatically generated by python-semantic-release (``827bd1c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/827bd1cf2db6cfcffdae98dbd6d24efac63d0cb6>`_)

- Merge pull request #27 from CycloneDX/feat/add-pipenv-support

FEATURE: Add `Pipfile.lock` (pipenv) support (``2c42e2a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2c42e2a616c07eec1f844b4fbc4e1e3b4a0815d8>`_)

- doc: updated README.md to include Pipfile.lock parsing

Signed-off-by: Paul Horton <phorton@sonatype.com> (``2c66834`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2c66834ee6aac75b3e810d13b5a3b41967043252>`_)

6.1.74 v0.6.2 (2021-10-11)

Fix

- fix: added ability to add tools in addition to this library when generating CycloneDX + plus fixes relating to multiple BOM instances

Signed-off-by: Paul Horton <phorton@sonatype.com> (``e03a25c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e03a25c3d2a1a0b711204bb26c7b898eadacdcb0>`_)

Unknown

- 0.6.2

Automatically generated by python-semantic-release (``e68fbc2`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e68fbc2ff5576fc1f5c0444f601c58f40f3cd917>`_)

- Merge branch 'main' of github.com:CycloneDX/cyclonedx-python-lib (``2bf2711`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2bf27119e7a1a3716706c28c3fb259496d0de6f1>`_)

6.1.75 v0.6.1 (2021-10-11)

Ci

- ci: update to deploy to pypi.org upon PR merge

Signed-off-by: Paul Horton <phorton@sonatype.com> (``04e86b5`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/04e86b54d71bf801511c728db949d622ae0c6fdc>`_)

Fix

- fix: better methods for checking if a Component is already represented in the BOM, and the ability to get the existing instance

Signed-off-by: Paul Horton <phorton@sonatype.com> (``5fee85f`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5fee85fc38376478a1a438d228c632a5d14f4740>`_)

Unknown

- 0.6.1

Automatically generated by python-semantic-release (``c530460`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c530460f504939d34e8c73066bfdd252dd95f090>`_)

- Merge branch 'main' of github.com:CycloneDX/cyclonedx-python-lib (``eb3a46b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/eb3a46b4365818dec08ea079f47e4abd75ebbd64>`_)

6.1.76 v0.6.0 (2021-10-11)

Feature

- feat: helper method for representing a File as a Component taking into account versioning for files as per https://github.com/CycloneDX/cyclonedx.org/issues/34

Signed-off-by: Paul Horton <phorton@sonatype.com> (``7e0fb3c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7e0fb3c7e32e08cb8667ad11461c7f8208dfdf7f>`_)

- feat: support for non-PyPi Components - PackageURL type is now definable when creating a Component

Signed-off-by: Paul Horton <phorton@sonatype.com> (``fde79e0`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fde79e02705bce216e62acd05056b6d2046cde22>`_)

Unknown

- 0.6.0

Automatically generated by python-semantic-release (``907cd2d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/907cd2d317f3cfd28febb450959938d09815b9c2>`_)

- Merge pull request #25 from CycloneDX/feat/additions-to-enable-integration-into-checkov

Support for representing File as Component (``63a86b0`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/63a86b05aa722078d57f143f35c1f5600396ec7a>`_)

6.1.77 v0.5.0 (2021-10-11)

Build

- build: updated dependencies, moved pdoc3 to a dev dependency

Signed-off-by: Paul Horton <phorton@sonatype.com> (``6a9947d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6a9947de1036b63804352e45c035d40658d3db01>`_)

Feature

- feat: add support for tool(s) that generated the SBOM

Signed-off-by: Paul Horton <phorton@sonatype.com> (``7d1e6ef`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7d1e6ef04d473407b9b4eefc2ef18e6723838f94>`_)

Fix

- fix: bumped a dependency version

Signed-off-by: Paul Horton <phorton@sonatype.com> (``efc1053`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/efc1053ec9ed3f57711f78f1eca181f7bff0c3bf>`_)

Unknown

- 0.5.0

Automatically generated by python-semantic-release (``a655d29`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a655d29ae9a93bdd72fee481d6a0ec8b71f6cce0>`_)

- Merge pull request #20 from CycloneDX/feat/additional-metadata

feat: add support for tool(s) that generated the SBOM (``b33cbf4`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b33cbf4cb40179e5710729b89d3c120e69448777>`_)

- fix for Pytho< 3.8 support in tests

Signed-off-by: Paul Horton <phorton@sonatype.com> (``c9b6019`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c9b6019609ae206ba965d0c4f7c06ffcf8835e1d>`_)

- ensure support for Python < 3.8

Signed-off-by: Paul Horton <phorton@sonatype.com> (``53a82cf`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/53a82cfbe7e828380c31b2441113f318d2a2c99e>`_)

- ensure support for Python < 3.8

Signed-off-by: Paul Horton <phorton@sonatype.com> (``2a9e56a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2a9e56a7e1e0235a06aa70f7750f1656f9305a8a>`_)

- doc: added documentation

Signed-off-by: Paul Horton <phorton@sonatype.com> (``cf13c68`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/cf13c6817552c0a6549ecd7131fdcd437ccc7210>`_)

- poetry(deps): bump zipp from 3.5.0 to 3.6.0

Bumps zipp from 3.5.0 to 3.6.0.

- Release notes
- Changelog
- Commits

---

updated-dependencies:

- dependency-name: zipp dependency-type: indirect update-type: version-update:semver-minor ...

Signed-off-by: dependabot[bot] <support@github.com> (``30f2547`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/30f254724b49c7596c58f11ef8f5a182706ef03a>`_)

- doc: bumped gh-action for publishing docs

Signed-off-by: Paul Horton <phorton@sonatype.com> (``ac70eee`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ac70eeed9325892ef9ae44b162d8a3ae43a435cc>`_)

- doc: added documentation to model/bom

Signed-off-by: Paul Horton <phorton@sonatype.com> (``fe98ada`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fe98ada121279f6119f3045abd737cc5b775a30f>`_)

- doc: formatting

Signed-off-by: Paul Horton <phorton@sonatype.com> (``1ad7fb1`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1ad7fb117acbec87def897f4dc549dc398decce6>`_)

- doc: added missing docstrings to allow documentation to generate

Signed-off-by: Paul Horton <phorton@sonatype.com> (``ed743d9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ed743d9b90904a6719309de85078657f9e4a48cd>`_)

- Merge pull request #10 from coderpatros/docs

Add initial doc generation and publishing (``7873ad9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7873ad9d3fed8c04b94999c21345ae4ca198e091>`_)

6.1.78 v0.4.1 (2021-09-27)

Build

- build: dependencies updated

Signed-off-by: Paul Horton <phorton@sonatype.com> (``0411826`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/04118263c2fed1241c4a9f38cc256542ba543d50>`_)

Fix

- fix: improved handling for `requirements.txt` content without pinned or declared versions

Signed-off-by: Paul Horton <phorton@sonatype.com> (``7f318cb`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7f318cb495ac1754029088cae1ef2574c58da2e5>`_)

Test

- test: additional tests around issue #8 which confirm level of support currently

Signed-off-by: Paul Horton <phorton@sonatype.com> (``bc54bed`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bc54bed79cbeb16dbfcb8c6aaea88d906fd8538a>`_)

- test: additional tests added to validate comments in requirements.txt and that hashes within requirements.txt are not currently supported

Signed-off-by: Paul Horton <phorton@sonatype.com> (``3a27d54`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3a27d546d56d5c5c27f77af716a5545723794294>`_)

Unknown

- 0.4.1

Automatically generated by python-semantic-release (``d5b7a2f`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d5b7a2fc731b29fd7a3f29fe3c94f14a98a82e69>`_)

- Merge pull request #15 from CycloneDX/fix/issue-14-requirements-unpinned-versions

fix: improved handling for `requirements.txt` content without pinned … (``f248015`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f248015ff9719dd0029f6267067356672f16f8c3>`_)

- Add initial doc generation and publishing

Signed-off-by: Patrick Dwyer <patrick.dwyer@owasp.org> (``cd1b558`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/cd1b558fe472895f9332d9844f99e652c14ec41e>`_)

6.1.79 v0.4.0 (2021-09-16)

Feature

- feat: support for localising vectors (i.e. stripping out any scheme prefix)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*b9e9e17*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b9e9e17ba1e2c1c9dfe551c61ad5152eebd829ab>`_)

- feat: helper methods for deriving Severity and SourceType

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*6a86ec2*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6a86ec27c13ff5e413c5a5f96d9b7671646f9388>`_)

Fix

- fix: removed print call

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*8806553*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/880655304c082a88d94d6d50c64d33ad931cc974>`_)

- fix: relaxed typing of parameter to be compatible with Python < 3.9

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*f9c7990*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f9c7990695119969c5055bc92a233030db999b84>`_)

- fix: removed print call

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*d272d2e*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d272d2ea7d3331bde0660bdc87a6ac3331ae0720>`_)

- fix: remove unused commented out code

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*ba4f285*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ba4f285fdbe124c28f7ea60310347cf896540125>`_)

Unknown

- 0.4.0

Automatically generated by python-semantic-release (``*f441413*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f441413668676c0435b173c01d612e9040d6f6db>`_)

6.1.80 v0.3.0 (2021-09-15)

Feature

- feat: adding support for extension schema that descriptions vulnerability disclosures

Signed-off-by: Paul Horton <phorton@sonatype.com> (``*d496695*` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d4966951ab6c0229171cfe97723421bb0302c4fc>`_)

Refactor

- refactor: moved Vulnerabilities to be nested inside the Component

Signed-off-by: Paul Horton <phorton@sonatype.com> (``8b4034d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8b4034da82a0c5e861161849ddb32c3806adfa0f>`_)

Test

- test: added test to confirm no Vulnerabilities are output for Schema Version 1.0 (not supported by schema)

Signed-off-by: Paul Horton <phorton@sonatype.com> (``d5aabcf`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d5aabcff8d46f635b3b74821d70fc279263c420c>`_)

Unknown

- 0.3.0

Automatically generated by python-semantic-release (``a5c3dab`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a5c3dab5818c183bd88385c7ad88e11eb34a0417>`_)

- Merge pull request #5 from CycloneDX/feat/support-schema-extension-vulnerability-1.0

FEATURE: add support for Vulnerability Disclosures (``6914272`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/69142723935199409f6bf91b68ecf1e91107f165>`_)

- doc: updated README to explain support for Vulnerability Disclosures

Signed-off-by: Paul Horton <phorton@sonatype.com> (``f477bf0`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f477bf03fc78cc2652e97cd77a3e7ab66306a39b>`_)

6.1.81 v0.2.0 (2021-09-14)

Feature

- feat: added helper method to return a PackageURL object representing a Component

Signed-off-by: Paul Horton <phorton@sonatype.com> (``367bef1`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/367bef11bb1a7ede3100acae39581e33d20fa7f5>`_)

Fix

- fix: whitespace on empty line removed

Signed-off-by: Paul Horton <phorton@sonatype.com> (``cfc952e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/cfc952eb5f3feb97a41b6c895657058429da3430>`_)

Unknown

- 0.2.0

Automatically generated by python-semantic-release (``866eda7`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/866eda764d01ee85778bea662c7556113121137e>`_)

- Merge pull request #4 from CycloneDX/feat/component-as-packageurl

fix: whitespace on empty line removed (``ddc37f3`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ddc37f395a1dbace39280a4f7b1074d954414f2d>`_)

- Merge branch 'main' of github.com:CycloneDX/cyclonedx-python-lib (``6142d2e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6142d2e3b9b655ebf95b59c93525ce8008851b34>`_)

6.1.82 v0.1.0 (2021-09-13)

Feature

- feat: add poetry support

Signed-off-by: Paul Horton <phorton@sonatype.com> (``f3ac42f`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f3ac42f298b8d093b0ac368993beba43c58c251a>`_)

Unknown

- 0.1.0

Automatically generated by python-semantic-release (``0da668f`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0da668f398bef2baee63b0d342063b6dc0eea71a>`_)

- Merge pull request #3 from CycloneDX/feat/poetry-lock-support

FEATURE: Adde poetry.lock parser support (``37ba7c6`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/37ba7c61a17881fc02119dcfd7b6e0a7cab48cbf>`_)

- feat(parser) - added support for parsing dependencies from poetry.lock files.

Signed-off-by: Paul Horton <phorton@sonatype.com> (``15bc553`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/15bc5539e2339581f80048a571ca632f17988530>`_)

- fix(parser) parsers were able to share state unexpectedly

Signed-off-by: Paul Horton <phorton@sonatype.com> (``dc59914`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/dc59914e961104d9fcd37822b172d798e68b6ebd>`_)

6.1.83 v0.0.11 (2021-09-10)

Fix

- fix(test): test was not updated for revised author statement

Signed-off-by: Paul Horton <phorton@sonatype.com> (``d1c9d37`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d1c9d379a1e92ee49aae8d133e2ad3e117054ec9>`_)

- fix(build): test failure and dependency missing

Fixed failing tests due to dependency on now removed VERSION file Added flake8 officially as a DEV dependency to poetry

Signed-off-by: Paul Horton <phorton@sonatype.com> (``9a2cfe9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9a2cfe94386b51acca44ae3bacae319b9b3c8f0d>`_)

- fix(build): removed artefacts associtated with non-poetry build

Tidied up project to remove items associated with non-Poetry build process. Also aligned a few references in README to new home of this project under CycloneDX.

Signed-off-by: Paul Horton <phorton@sonatype.com> (``f9119d4`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f9119d49e462cf1f7ccca9c50af2936f8962fd6d>`_)

Unknown

- 0.0.11

Automatically generated by python-semantic-release (``1c0aa71`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1c0aa716b36e1305b7a3a2b9e2dfd6e5c6ac0011>`_)

- Merge pull request #2 from CycloneDX/fix/tidy-up-build-remove-pip

fix(build): removed artefacts associated with non-poetry build (``b7de7b3`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b7de7b3c9ba2c8c824d898ee994169b66b78b07a>`_)

6.1.84 v0.0.10 (2021-09-08)

Fix

- fix: add in pypi badge (``6098c36`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6098c36715b2459d7b04ced5ba6294437576e481>`_)

Unknown

- 0.0.10

Automatically generated by python-semantic-release (``245d809`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/245d809c3918d023ae58af2fb352f14912be091c>`_)

6.1.85 v0.0.9 (2021-09-08)

Fix

- fix: additional info to poetry, remove circleci (``2fcfa5a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2fcfa5ac3a7d9d7f372be6d69e1c616b551877df>`_)

Unknown

- 0.0.9

Automatically generated by python-semantic-release (``e4a90cf`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e4a90cfc46db3284e1f3e53f6555405fc14dc654>`_)

- Merge branch 'main' of github.com:CycloneDX/cyclonedx-python-lib into main (``69aaba5`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/69aaba5f941cbffc40b47d18c6f9dd9dd754b57b>`_)

6.1.86 v0.0.8 (2021-09-08)

Fix

- fix: initial release to pypi, tell poetry to include cyclonedx package (``a030177`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a030177cb1a370713c4438b13b7520ef6afd19f6>`_)

Unknown

- 0.0.8

Automatically generated by python-semantic-release (`fc3f24c` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/fc3f24c13938948c4786ecf8ace3fc241c0f458e>`_)

- Merge branch 'main' of github.com:CycloneDX/cyclonedx-python-lib into main (``da2d18c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/da2d18cd60a781bf097e563466bda0d3e51b9e8f>`_)

6.1.87 v0.0.7 (2021-09-08)

Fix

- fix: release with full name (``4c620ed`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/4c620ed053aac8c31343b1ca84ca56912b762ab2>`_)

Unknown

- 0.0.7

Automatically generated by python-semantic-release (``19943e8`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/19943e8287bbe67031cada6f5377d438f2b033c1>`_)

6.1.88 v0.0.6 (2021-09-08)

Fix

- fix: initial release to pypi (``99687db`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/99687dbec1389bf323bb625bfb707306aa3b8d1a>`_)

Unknown

- 0.0.6

Automatically generated by python-semantic-release (``98ad249`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/98ad24950dbb5f5b08db41e1bb4e359f8f0b8b49>`_)

- Switch to using action (``cce468a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/cce468a7004d848ddbaab4affa392bd2f74414dd>`_)

6.1.89 v0.0.5 (2021-09-08)

Unknown

- 0.0.5

Automatically generated by python-semantic-release (``9bf4b9a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9bf4b9a29cc4b0bbdf5771ffc22b918a6081a0a1>`_)

- Merge branch 'main' of github.com:CycloneDX/cyclonedx-python-lib into main (``eeec0bb`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/eeec0bba7d0a615f8384caa50ed95c2240b5a951>`_)
- Try this on for size (``aa93310`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/aa93310830a86aa441337be34081c46d9475384c>`_)

6.1.90 v0.0.4 (2021-09-08)

Unknown

- 0.0.4

Automatically generated by python-semantic-release (``b16d6c5`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b16d6c59495de396c73dfe1ffabcbfd325dfa619>`_)

- Use python3 to install (``4c810e1`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/4c810e16b1a93afb923652f66e77ee08ff0ffd49>`_)

6.1.91 v0.0.3 (2021-09-08)

Unknown

- 0.0.3

Automatically generated by python-semantic-release (``05306ee`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/05306ee235df1d7aa662c9323e6186cc3d1129dc>`_)

- Merge branch 'main' of github.com:CycloneDX/cyclonedx-python-lib into main (``f1d120c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f1d120c5dca530424dd79b3303458cc0adbc28de>`_)
- Bump up version of poetry (``89db268`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/89db2689bbdb94f2f290abe1bf721b163d75001e>`_)

6.1.92 v0.0.2 (2021-09-08)

Unknown

- 0.0.2

Automatically generated by python-semantic-release (``e15dec6`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e15dec696bd88d00f5f5fdce74cb407bc65a42e2>`_)

- Remove check for push (``71b1270`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/71b12709f0fb55852cbb030669a80a5ebd2f2e92>`_)
- Manual deploy workflow (``9b4ac33`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9b4ac335becf7e7b83cd3fa619c8975b6335f5eb>`_)
- License headers, OWASP etc… (``559b8d2`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/559b8d227e52b6798a71149c87f4090ea1244c85>`_)
- Fixed unit tests pinned to a VERISON. (``5d907d5`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5d907d58e57f2eb7731047a51a88104cb07c1796>`_)
- Bump to version 0.0.2 (``1050839`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/105083951dc93f28a4816c0c699af7db7f2789d9>`_)
- Implemented writing SBOM to a file. (``74f4153`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/74f4153d84c3bbdb875eac679fe933b777f90f18>`_)
- Updated badge in README to include Python 3.6+ support. (``0a5903c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0a5903c56971a19172fe904f02836c5c5e2262db>`_)
- Removed print() statement accidentally left in. (``22965a7`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/22965a707de6db7bb08721809035562be72c69d5>`_)
- Merge pull request #1 from sonatype-nexus-community/features/initial-port-of-v1.1-generation-from-jake

Initial port of library code to new library (``2f2634b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2f2634b86612b4f0d2142b09f3aece588937fcaa>`_)

- Added license headers to all source files. Added classifiers for Python version to setup.py. (``bb6bb24`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bb6bb24440996257ce609b0f399f930153b65e8e>`_)
- Renamed model file to not reference CycloneDX as the models are agnostic on purpose. (``03d03ed`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/03d03edfca7bed56d21733120cb5b002a32bb466>`_)
- Forgot to add updated poetry.lock file relfecting Python 3.6+ support (``5d3d491`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/5d3d49184039a2f41411cd96d5dfcf1544fab05f>`_)
- Updated project to state support from Python v3.6+ (``619ee1d`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/619ee1dfc23f7220a1941c3fa5068761346c84cb>`_)
- Adding Python 3.6 support for test & CI. (``daa12ba`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/daa12ba8925128da040cf836bc3f16a2126e9091>`_)
- Fixing CircleCI config. (``a446f4c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a446f4cb197fd40a3065a372108c1719cde91136>`_)
- Fixes to GitHub actions. (``d2aa277`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/d2aa277bce954100adad42e33c095bc1f9ce23cd>`_)
- Disabled Py3.6 checks and added flake8. (``8c01da3`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8c01da3d8f6038fb24df07ab3fb0945c79893e9f>`_)
- Attempt to fix CI's for multiple Python environments. (``affb6b2`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/affb6b2dc7afeaff5b5cd0a1d4f65678394a2ff7>`_)

- Added support for Python versions 3.7+ (``ae24ba9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ae24ba9c26ddf4ef91937e8489b1894a986724de>`_)
- Added missing ENV var for GH actions. (``c750ec6`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c750ec62411c6d4473d3cc0a33dc96f90a443cef>`_)
- Missed wrapping a coverage command with poetry. (``3c74c82`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3c74c822445e5aeaaa387c8e5522ca8cd841cfd8>`_)
- Added poetry virtualenv caching + wrapped tox and coverage with poetry to ensure they run in the poetry venv. (``780e3df`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/780e3dfa043957174e1f79cf450d1ee69d6530d3>`_)
- Fixed typo in Github action. (``3953675`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/395367531e7a00c086e723a78d059e6016fb242e>`_)
- Correction: Supported Python version in setup.py (``2f4917b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/2f4917ba81f8ddba994a2c5012303bccb307a419>`_)
- Updated poetry dependencies and configuration. (``75041e5`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/75041e51ff684853d7c2b94e5a722a4ec14043fc>`_)
- Initial draft GitHub actions being added. (``e2403e8`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e2403e8c4194be6bee70a58ef86d9acec6de5dbb>`_)
- Added Poetry supprot. (``e9a67f8`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e9a67f8a405b6c664d2b91bd4966a8ade9902d40>`_)
- Addressing issues reported by flake8. (``3ad394c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3ad394c14d9cbf3e706f4fe47b6f83938576a2ac>`_)
- Refactored output classes to use multiple inheritance allowing a single place to define which schema version support various attributes and elements. (``95c5b38`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/95c5b389bb5c8c358420aaf5c62694dcabe663ce>`_)
- Updated README to reflect support for author. (``bff5954`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bff5954f70967f3605fa6226a223590b89e07313>`_)
- Skeleton support for 'author' + v1.1 and v1.0 for JSON added (along with tests). (``e987f35`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e987f357314199442ed2c5823575833915dfccb1>`_)
- Corrected typo in README (``0d2c355`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/0d2c35519374b4efddf399dd519e5a1443a56692>`_)
- Updated README to include a summary of the support this library provides across the different schema versions. (``34f421f`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/34f421f4076d16c30ddf291f5c1866c1b623258a>`_)
- Initial support for V1.0 and V1.1 in XML output format. (``37f6b00`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/37f6b00b7e354b76a9f8f72ed2c1004a0e728319>`_)
- Added 'serialNumber' to SBOMs (JSON and XML). (``50e3c75`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/50e3c7546b92e3241feefa6dea0fbfa9c1145843>`_)
- Added a bunch more content to the README to explain how the library can be used. (``bb41dc6`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/bb41dc6d333f59025aae97c602cbe41343645b20>`_)
- Added metadata initial support to JSON output format. (``8c5590f`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/8c5590fd3c5c59de9a5b6cf49005f4c6e444265d>`_)
- Addition of simple 'metadata' element for XML SBOM's. (``f9e9773`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/f9e97733b0cc57bbb71341b4ced4ccc8f09b7f28>`_)

- Added initial JSON outputter and associated tests. (``3e1f5ec`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/3e1f5ec9354a779adf44129656a1ccdcffadee6d>`_)
- Fix to generate HTML coverage reports and stash in CircleCI builds. (``dd88603`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/dd886032b92d491f462d62f269f3df7ed823d436>`_)
- Added HTML coverage report. (``ce700e5`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ce700e5bdff7ce4a8bd5614239b129e59afe2908>`_)
- Missed coverage as a dependency for testing. (``01643d6`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/01643d67f73ec8ee35884d0bcc15c892649f6b72>`_)
- Added coverage reporting for tests (``c34b1a6`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/c34b1a63fd7958d2b1060ba51054a55b57228549>`_)
- Added first tests for XML SBOM generation (v1.3 and v1.2). (``cb4337a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/cb4337a1cb14ee62471140add8954dd7c5b6b314>`_)
- WIP: Starting to generate XML output for BOMs (``35bdfca`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/35bdfca4fc01cdb3fa7ab6fb37b1c05eaa7189ec>`_)
- Updated CircleCI config to run tox. Fixed fomratting in tests. (``9a56230`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/9a5623098ff712df0cefbd2327e8058f9ac74e17>`_)
- Rebasing from main. (``822ab8b`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/822ab8b43a06bf1712d134d44acb136e70134c05>`_)
- Initial skeleton tests for output genereation. (``a614f3e`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/a614f3e9cc6210a25daff79e4ec428f15221cc1e>`_)
- pretty badge (``60e975c`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/60e975c12cdf6c15c9e38585becaf53850609d67>`_)
- initial CI for discussion (``7e88cd5`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/7e88cd5920480cd6bde4e72b8b85314242964013>`_)
- Added a little more information to the README. (``460c624`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/460c62487e66df750a99e10a62bf19bf0baf2e76>`_)
- Fixed issue reported by Flake8. Ensuring tests run on PY 3.9. (``cce130f`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/cce130f53a7c73554015ce672cbe8799e863e64b>`_)
- Basic structure without any output generation available (very basic Component definition). (``6ac5dc2`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/6ac5dc29fb4bc52f66698966e0b570588621be72>`_)
- Added tox config with flake8 and py3.9 support. (``1def201`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/1def2015d3aad4b58980d9b86cca840f19ac4ee6>`_)
- Initially added skeleton packaging structure and official CycloneDX schemas. (``ac519c9`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/ac519c9a21bc8e4a75927868f32f29febc648509>`_)
- Added inital blank README prior to branching for initial work. (``b175f6a`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/b175f6a9178c510cfa14b5d2788feecfd65d8e94>`_)
- Added inital blank README prior to branching for initial work. (``e8b5d48`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/e8b5d4802079f92da106b8e0a68f9311c328a656>`_)
- Initial commit (``62353b0`` <https://github.com/CycloneDX/cyclonedx-python-lib/commit/62353b0ce57f797bcb9dfd97871e886db8269478>`_)

CHAPTER

SEVEN

API REFERENCE

This page contains auto-generated API reference documentation[1].

7.1 cyclonedx

Python library for CycloneDX

7.1.1 Subpackages

cyclonedx.exception

Exceptions that are specific to the CycloneDX library implementation.

Submodules

cyclonedx.exception.factory

Exceptions relating to specific conditions that occur when factoring a model.

Module Contents

exception cyclonedx.exception.factory.**CycloneDxFactoryException**

Bases: *cyclonedx.exception.CycloneDxException*

Base exception that covers all exceptions that may be thrown during model factoring..

class args

add_note()

Exception.add_note(note) – add a note to the exception

with_traceback()

Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

[1] Created with sphinx-autoapi

exception cyclonedx.exception.factory.**LicenseChoiceFactoryException**

Bases: *CycloneDxFactoryException*

Base exception that covers all LicenseChoiceFactory exceptions.

class args

add_note()

Exception.add_note(note) – add a note to the exception

with_traceback()

Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception cyclonedx.exception.factory.**InvalidSpdxLicenseException**

Bases: *LicenseChoiceFactoryException*

Thrown when an invalid SPDX License is provided.

class args

add_note()

Exception.add_note(note) – add a note to the exception

with_traceback()

Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception cyclonedx.exception.factory.**LicenseFactoryException**

Bases: *CycloneDxFactoryException*

Base exception that covers all LicenseFactory exceptions.

class args

add_note()

Exception.add_note(note) – add a note to the exception

with_traceback()

Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception cyclonedx.exception.factory.**InvalidLicenseExpressionException**

Bases: *LicenseFactoryException*

Thrown when an invalid License expressions is provided.

class args

add_note()

Exception.add_note(note) – add a note to the exception

with_traceback()

Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

cyclonedx.exception.model

Exceptions relating to specific conditions that occur when modelling CycloneDX BOM.

Module Contents

Classes

CycloneDxModelException	Base exception that covers all exceptions that may be thrown during model creation.
InvalidLocaleTypeException	Raised when the supplied locale does not conform to ISO-639 specification.
InvalidNistQuantumSecurityLevelException	Raised when an invalid value is provided for an NIST Quantum Security Level
InvalidOmniBorIdException	Raised when a supplied value for an OmniBOR ID does not meet the format requirements
InvalidRelatedCryptoMaterialSizeException	Raised when the supplied size of a Related Crypto Material is negative.
InvalidSwhidException	Raised when a supplied value for an Swhid does not meet the format requirements
InvalidUriException	Raised when a *str* is provided that needs to be a valid URI, but isn't.
MutuallyExclusivePropertiesException	Raised when mutually exclusive properties are provided.
NoPropertiesProvidedException	Raised when attempting to construct a model class and providing NO values (where all properites are defined as
UnknownComponentDependencyException	Exception raised when a dependency has been noted for a Component that is NOT a Component BomRef in this Bom.
UnknownHashTypeException	Exception raised when we are unable to determine the type of hash from a composite hash string.
LicenseExpressionAlongWithOthersException	Exception raised when a LicenseExpression was detected along with other licenses.

class cyclonedx.exception.model.**CycloneDxModelException**

Bases: *cyclonedx.exception.CycloneDxException*

Base exception that covers all exceptions that may be thrown during model creation.

class cyclonedx.exception.model.**InvalidLocaleTypeException**

Bases: *CycloneDxModelException*

Raised when the supplied locale does not conform to ISO-639 specification.

Good examples:

- en
- en-US
- en-GB
- fr
- fr-CA

The language code MUST be lowercase. If the country code is specified, the country code MUST be upper case. The language code and country code MUST be separated by a minus sign.

class cyclonedx.exception.model.**InvalidNistQuantumSecurityLevelException**

Bases: *CycloneDxModelException*

Raised when an invalid value is provided for an NIST Quantum Security Level as defined at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/ evaluation-criteria/security-(evaluation-criteria).

class cyclonedx.exception.model.**InvalidOmniBorIdException**

Bases: *CycloneDxModelException*

Raised when a supplied value for an OmniBOR ID does not meet the format requirements as defined at https://www.iana.org/assignments/uri-schemes/prov/gitoid.

class cyclonedx.exception.model.**InvalidRelatedCryptoMaterialSizeException**

Bases: *CycloneDxModelException*

Raised when the supplied size of a Related Crypto Material is negative.

class cyclonedx.exception.model.**InvalidSwhidException**

Bases: *CycloneDxModelException*

Raised when a supplied value for an Swhid does not meet the format requirements as defined at https://docs.softwareheritage.org/devel/swh-model/persistent-identifiers.html.

class cyclonedx.exception.model.**InvalidUriException**

Bases: *CycloneDxModelException*

Raised when a *str* is provided that needs to be a valid URI, but isn't.

class cyclonedx.exception.model.**MutuallyExclusivePropertiesException**

Bases: *CycloneDxModelException*

Raised when mutually exclusive properties are provided.

class cyclonedx.exception.model.**NoPropertiesProvidedException**

Bases: *CycloneDxModelException*

Raised when attempting to construct a model class and providing NO values (where all properites are defined as Optional, but at least one is required).

class cyclonedx.exception.model.**UnknownComponentDependencyException**

Bases: *CycloneDxModelException*

Exception raised when a dependency has been noted for a Component that is NOT a Component BomRef in this Bom.

class cyclonedx.exception.model.**UnknownHashTypeException**

Bases: *CycloneDxModelException*

Exception raised when we are unable to determine the type of hash from a composite hash string.

class cyclonedx.exception.model.**LicenseExpressionAlongWithOthersException**

Bases: *CycloneDxModelException*

Exception raised when a LicenseExpression was detected along with other licenses. If a LicenseExpression exists, than it must stand alone.

See https://github.com/CycloneDX/specification/pull/205

cyclonedx.exception.output

Exceptions that are for specific error scenarios during the output of a Model to a SBOM.

Module Contents

Classes

BomGenerationErrorException	Raised if there is an unknown error.
FormatNotSupportedException	Exception raised when attempting to output a BOM to a format not supported in the requested version.

class cyclonedx.exception.output.**BomGenerationErrorException**

Bases: *cyclonedx.exception.CycloneDxException*

Raised if there is an unknown error.

class cyclonedx.exception.output.**FormatNotSupportedException**

Bases: *cyclonedx.exception.CycloneDxException*

Exception raised when attempting to output a BOM to a format not supported in the requested version.

For example, JSON is not supported prior to 1.2.

cyclonedx.exception.serialization

Exceptions relating to specific conditions that occur when (de)serializing/(de)normalizing CycloneDX BOM.

Module Contents

exception cyclonedx.exception.serialization.**CycloneDxSerializationException**

Bases: *cyclonedx.exception.CycloneDxException*

Base exception that covers all exceptions that may be thrown during model serializing/normalizing.

class args

add_note()

Exception.add_note(note) – add a note to the exception

with_traceback()

Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception cyclonedx.exception.serialization.**CycloneDxDeserializationException**

Bases: *cyclonedx.exception.CycloneDxException*

Base exception that covers all exceptions that may be thrown during model deserializing/denormalizing.

class args

add_note()

Exception.add_note(note) – add a note to the exception

with_traceback()

Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception cyclonedx.exception.serialization.**SerializationOfUnsupportedComponentTypeException**

Bases: *CycloneDxSerializationException*

Raised when attempting serializing/normalizing a *cyclonedx.model.component.Component* to a *cyclonedx.schema.schema.BaseSchemaVersion* which does not support that *cyclonedx.model.component.ComponentType* .

class args

add_note()

Exception.add_note(note) – add a note to the exception

with_traceback()

Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception cyclonedx.exception.serialization.**SerializationOfUnexpectedValueException**

Bases: *CycloneDxSerializationException*, ValueError

Raised when attempting serializing/normalizing a type that is not expected there.

class args

add_note()

Exception.add_note(note) – add a note to the exception

with_traceback()

Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

Package Contents

exception cyclonedx.exception.**CycloneDxException**

Bases: Exception

Root exception thrown by this library.

class args

add_note()

Exception.add_note(note) – add a note to the exception

with_traceback()

Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception cyclonedx.exception.**MissingOptionalDependencyException**

Bases: *CycloneDxException*

Validation did not happen, due to missing dependencies.

class args

add_note()

Exception.add_note(note) – add a note to the exception

with_traceback()

Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

cyclonedx.factory

Factories used in this library.

Submodules

cyclonedx.factory.license

Module Contents

Classes

LicenseFactory	Factory for *cyclonedx.model.license.License*.

class cyclonedx.factory.license.**LicenseFactory**

Factory for *cyclonedx.model.license.License*.

make_from_string(*value: str*, ***, *license_text:* cyclonedx.model.AttachedText | *None* = *None*, *license_url:* cyclonedx.model.XsUri | *None* = *None*, *license_acknowledgement:* cyclonedx.model.license.LicenseAcknowledgement | *None* = *None*) → cyclonedx.model.license.License

Make a *cyclonedx.model.license.License* from a string.

make_with_expression(*expression: str*, ***, *acknowledgement:* cyclonedx.model.license.LicenseAcknowledgement | *None* = *None*) → *cyclonedx.model.license.LicenseExpression*

Make a *cyclonedx.model.license.LicenseExpression* with a compound expression.

Utilizes *cyclonedx.spdx.is_compound_expression()*.

Raises

InvalidLicenseExpressionException – if param *value* is not known/supported license expression

make_with_id(*spdx_id: str*, ***, *text:* cyclonedx.model.AttachedText | *None* = *None*, *url:* cyclonedx.model.XsUri | *None* = *None*, *acknowledgement:* cyclonedx.model.license.LicenseAcknowledgement | *None* = *None*) → *cyclonedx.model.license.DisjunctiveLicense*

Make a *cyclonedx.model.license.DisjunctiveLicense* from an SPDX-ID.

Raises

InvalidSpdxLicenseException – if param *spdx_id* was not known/supported SPDX-ID

make_with_name(*name: str*, ***, *text:* cyclonedx.model.AttachedText | *None* = *None*, *url:* cyclonedx.model.XsUri | *None* = *None*, *acknowledgement:* cyclonedx.model.license.LicenseAcknowledgement | *None* = *None*) → *cyclonedx.model.license.DisjunctiveLicense*

Make a *cyclonedx.model.license.DisjunctiveLicense* with a name.

cyclonedx.model

Uniform set of models to represent objects within a CycloneDX software bill-of-materials.

You can either create a *cyclonedx.model.bom.Bom* yourself programmatically, or generate a *cyclonedx.model.bom.Bom* from a *cyclonedx.parser.BaseParser* implementation.

Submodules

cyclonedx.model.bom

Module Contents

Classes

BomMetaData	This is our internal representation of the metadata complex type within the CycloneDX standard.
Bom	This is our internal representation of a bill-of-materials (BOM).

class cyclonedx.model.bom.**BomMetaData**(*, *tools: Iterable[*cyclonedx.model.Tool*] | None = None*, *authors: Iterable[*cyclonedx.model.contact.OrganizationalContact*] | None = None*, *component:* cyclonedx.model.component.Component *| None = None*, *supplier:* cyclonedx.model.contact.OrganizationalEntity *| None = None*, *licenses: Iterable[cyclonedx.model.license.License] | None = None*, *properties: Iterable[*cyclonedx.model.Property*] | None = None*, *timestamp: datetime.datetime | None = None*, *manufacturer:* cyclonedx.model.contact.OrganizationalEntity *| None = None*, *manufacture:* cyclonedx.model.contact.OrganizationalEntity *| None = None*)

This is our internal representation of the metadata complex type within the CycloneDX standard.

Note: See the CycloneDX Schema for Bom metadata: https://cyclonedx.org/docs/1.5/#type_metadata

property timestamp: datetime.datetime

The date and time (in UTC) when this BomMetaData was created.

Returns:
datetime instance in UTC timezone

property tools: SortedSet[*Tool*]

Tools used to create this BOM.

Returns:
Set of *Tool* objects.

property authors: SortedSet[*OrganizationalContact*]

The person(s) who created the BOM.

Authors are common in BOMs created through manual processes.

BOMs created through automated means may not have authors.

Returns:
Set of *OrganizationalContact*

property component: *cyclonedx.model.component.Component* | **None**

The (optional) component that the BOM describes.

Returns:
cyclonedx.model.component.Component instance for this Bom Metadata.

property manufacture: *cyclonedx.model.contact.OrganizationalEntity* | **None**

The organization that manufactured the component that the BOM describes.

Returns:
OrganizationalEntity if set else *None*

property manufacturer: *cyclonedx.model.contact.OrganizationalEntity* | **None**

The organization that created the BOM. Manufacturer is common in BOMs created through automated processes. BOMs created through manual means may have *@.authors* instead.

Returns:
OrganizationalEntity if set else *None*

property supplier: *cyclonedx.model.contact.OrganizationalEntity* | **None**

The organization that supplied the component that the BOM describes.

The supplier may often be the manufacturer, but may also be a distributor or repackager.

Returns:
OrganizationalEntity if set else *None*

property licenses: *cyclonedx.model.license.LicenseRepository*

A optional list of statements about how this BOM is licensed.

Returns:
Set of *LicenseChoice*

property properties: SortedSet[*Property*]

Provides the ability to document properties in a key/value store. This provides flexibility to include data not officially supported in the standard without having to use additional namespaces or create extensions.

Property names of interest to the general public are encouraged to be registered in the CycloneDX Property Taxonomy - https://github.com/CycloneDX/cyclonedx-property-taxonomy. Formal registration is OPTIONAL.

Return:
Set of *Property*

class cyclonedx.model.bom.**Bom**(*, *components: Iterable[*cyclonedx.model.component.Component*]* | *None* = *None*, *services: Iterable[*cyclonedx.model.service.Service*]* | *None* = *None*, *external_references: Iterable[*cyclonedx.model.ExternalReference*]* | *None* = *None*, *serial_number: uuid.UUID* | *None* = *None*, *version: int* = *1*, *metadata:* BomMetaData | *None* = *None*, *dependencies: Iterable[*cyclonedx.model.dependency.Dependency*]* | *None* = *None*, *vulnerabilities: Iterable[*cyclonedx.model.vulnerability.Vulnerability*]* | *None* = *None*, *properties: Iterable[*cyclonedx.model.Property*]* | *None* = *None*)

This is our internal representation of a bill-of-materials (BOM).

Once you have an instance of *cyclonedx.model.bom.Bom*, you can pass this to an instance of *cyclonedx.output.BaseOutput* to produce a CycloneDX document according to a specific schema version and format.

property serial_number: uuid.UUID

Unique UUID for this BOM

Returns:
UUID instance *UUID* instance

property version: int

property metadata: *BomMetaData*

Get our internal metadata object for this Bom.

Returns:
Metadata object instance for this Bom.

Note: See the CycloneDX Schema for Bom metadata: https://cyclonedx.org/docs/1.3/#type_metadata

---

property components: SortedSet[*Component*]

Get all the Components currently in this Bom.

Returns:
Set of *Component* in this Bom

property services: SortedSet[*Service*]

Get all the Services currently in this Bom.

Returns:
Set of *Service* in this BOM

property external_references: SortedSet[*ExternalReference*]

Provides the ability to document external references related to the BOM or to the project the BOM describes.

Returns:
Set of *ExternalReference*

property dependencies: SortedSet[*Dependency*]

property properties: SortedSet[*Property*]

Provides the ability to document properties in a name/value store. This provides flexibility to include data not officially supported in the standard without having to use additional namespaces or create extensions. Property names of interest to the general public are encouraged to be registered in the CycloneDX Property Taxonomy - https://github.com/CycloneDX/cyclonedx-property-taxonomy. Formal registration is OPTIONAL.

Return:
Set of *Property*

property vulnerabilities: SortedSet[*Vulnerability*]

Get all the Vulnerabilities in this BOM.

Returns:
Set of *Vulnerability*

get_component_by_purl(*purl: packageurl.PackageURL | None*) → *cyclonedx.model.component.Component* | None

Get a Component already in the Bom by its PURL

Args:

purl:
An instance of *packageurl.PackageURL* to look and find *Component*.

Returns:
Component or *None*

`get_urn_uuid()` → str

Get the unique reference for this Bom.

Returns:
URN formatted UUID that uniquely identified this Bom instance.

`has_component(component: cyclonedx.model.component.Component)` → bool

Check whether this Bom contains the provided Component.

Args:

component:
The instance of *cyclonedx.model.component.Component* to check if this Bom contains.

Returns:
bool - *True* if the supplied Component is part of this Bom, *False* otherwise.

`get_vulnerabilities_for_bom_ref(bom_ref: cyclonedx.model.bom_ref.BomRef)` → SortedSet[*Vulnerability*]

Get all known Vulnerabilities that affect the supplied bom_ref.

Args:
bom_ref: *BomRef*

Returns:
SortedSet of *Vulnerability*

`has_vulnerabilities()` → bool

Check whether this Bom has any declared vulnerabilities.

Returns:
bool - *True* if this Bom has at least one Vulnerability, *False* otherwise.

`register_dependency(target: cyclonedx.model.dependency.Dependable, depends_on: Iterable[cyclonedx.model.dependency.Dependable] | None = None)` → None

`urn()` → str

`validate()` → bool

Perform data-model level validations to make sure we have some known data integrity prior to attempting output of this *Bom*

Returns:
bool

cyclonedx.model.bom_ref

Module Contents

Classes

BomRef	An identifier that can be used to reference objects elsewhere in the BOM.

class cyclonedx.model.bom_ref.**BomRef**(*value: str | None = None*)

An identifier that can be used to reference objects elsewhere in the BOM.

This copies a similar pattern used in the CycloneDX PHP Library.

Note: See https://github.com/CycloneDX/cyclonedx-php-library/blob/master/docs/dev/decisions/BomDependencyDataModel.md

property value: str | None

cyclonedx.model.component

Module Contents

Classes

Commit	Our internal representation of the *commitType* complex type.
ComponentEvidence	Our internal representation of the *componentEvidenceType* complex type.
ComponentScope	Enum object that defines the permissable 'scopes' for a Component according to the CycloneDX schema.
ComponentType	Enum object that defines the permissible 'types' for a Component according to the CycloneDX schema.
Diff	Our internal representation of the *diffType* complex type.
PatchClassification	Enum object that defines the permissible `patchClassification`s.
Patch	Our internal representation of the *patchType* complex type.
Pedigree	Our internal representation of the *pedigreeType* complex type.
Swid	Our internal representation of the *swidType* complex type.
OmniborId	Helper class that allows us to perform validation on data strings that must conform to
Swhid	Helper class that allows us to perform validation on data strings that must conform to
Component	This is our internal representation of a Component within a Bom.

class cyclonedx.model.component.**Commit**(*, *uid: str | None = None*, *url:* cyclonedx.model.XsUri | *None = None*, *author:* cyclonedx.model.IdentifiableAction | *None = None*, *committer:* cyclonedx.model.IdentifiableAction | *None = None*, *message: str | None = None*)

Our internal representation of the *commitType* complex type.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_commitType

property uid: str | None

A unique identifier of the commit. This may be version control specific. For example, Subversion uses revision numbers whereas git uses commit hashes.

Returns:
str if set else *None*

property url: *cyclonedx.model.XsUri* | None

The URL to the commit. This URL will typically point to a commit in a version control system.

Returns:
XsUri if set else *None*

property author: *cyclonedx.model.IdentifiableAction* | None

The author who created the changes in the commit.

Returns:
IdentifiableAction if set else *None*

property committer: *cyclonedx.model.IdentifiableAction* | None

The person who committed or pushed the commit

Returns:
IdentifiableAction if set else *None*

property message: str | None

The text description of the contents of the commit.

Returns:
str if set else *None*

class cyclonedx.model.component.**ComponentEvidence**(*, *licenses: Iterable[cyclonedx.model.license.License] | None = None*, *copyright: Iterable[*cyclonedx.model.Copyright*] | None = None*)

Our internal representation of the *componentEvidenceType* complex type.

Provides the ability to document evidence collected through various forms of extraction or analysis.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_componentEvidenceType

property licenses: *cyclonedx.model.license.LicenseRepository*

Optional list of licenses obtained during analysis.

Returns:
Set of *LicenseChoice*

property copyright: SortedSet[*Copyright*]

Optional list of copyright statements.

Returns:
Set of *Copyright*

class cyclonedx.model.component.**ComponentScope**

Bases: str, enum.Enum

Enum object that defines the permissable 'scopes' for a Component according to the CycloneDX schema.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.3/#type_scope

REQUIRED = 'required'

OPTIONAL = 'optional'

EXCLUDED = 'excluded'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

`islower()`

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

`isnumeric()`

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

`isprintable()`

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

`isspace()`

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

`istitle()`

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

`isupper()`

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

`join()`

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

`ljust()`

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

`lower()`

Return a copy of the string converted to lowercase.

`lstrip()`

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

`partition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

rfind()

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

rindex()

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

rjust()

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

rpartition()

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

rsplit()

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

`translate()`

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

`upper()`

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.component.**ComponentType**

Bases: str, enum.Enum

Enum object that defines the permissible 'types' for a Component according to the CycloneDX schema.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.3/#type_classification

---

APPLICATION = 'application'

CONTAINER = 'container'

CRYPTOGRAPHIC_ASSET = 'cryptographic-asset'

DATA = 'data'

DEVICE = 'device'

DEVICE_DRIVER = 'device-driver'

FILE = 'file'

FIRMWARE = 'firmware'

FRAMEWORK = 'framework'

LIBRARY = 'library'

MACHINE_LEARNING_MODEL = 'machine-learning-model'

OPERATING_SYSTEM = 'operating-system'

PLATFORM = 'platform'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

`count()`

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

`encode()`

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

`endswith()`

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

`expandtabs()`

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

`find()`

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`format()`

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

`format_map()`

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

`index()`

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`isalnum()`

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

`join()`

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

`ljust()`

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

`lower()`

Return a copy of the string converted to lowercase.

`lstrip()`

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

`partition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

`removeprefix()`

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

`removesuffix()`

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

`replace()`

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

strip()

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

swapcase()

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.component.**Diff**(*, *text:* cyclonedx.model.AttachedText | *None* = *None*, *url:* cyclonedx.model.XsUri | *None* = *None*)

Our internal representation of the *diffType* complex type.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_diffType

property text: *cyclonedx.model.AttachedText* | **None**

Specifies the optional text of the diff.

Returns:
AttachedText if set else *None*

property url: *cyclonedx.model.XsUri* | **None**

Specifies the URL to the diff.

Returns:
XsUri if set else *None*

class cyclonedx.model.component.**PatchClassification**

Bases: str, enum.Enum

Enum object that defines the permissible `patchClassification`s.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_patchClassification

BACKPORT = 'backport'

CHERRY_PICK = 'cherry-pick'

MONKEY = 'monkey'

UNOFFICIAL = 'unofficial'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

rfind()

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

rindex()

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

rjust()

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

rpartition()

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

rsplit()

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

`translate()`

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

`upper()`

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.component.**Patch**(*, *type:* PatchClassification, *diff:* Diff | *None* = *None*, *resolves:* *Iterable[*cyclonedx.model.issue.IssueType*]* | *None* = *None*)

Our internal representation of the *patchType* complex type.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_patchType

property type: *PatchClassification*

Specifies the purpose for the patch including the resolution of defects, security issues, or new behavior or functionality.

Returns:
PatchClassification

property diff: *Diff* | **None**

The patch file (or diff) that show changes.

Note: Refer to https://en.wikipedia.org/wiki/Diff.

Returns:
Diff if set else *None*

property resolves: **SortedSet[***IssueType***]**

Optional list of issues resolved by this patch.

Returns:
Set of *IssueType*

class cyclonedx.model.component.**Pedigree**(*, *ancestors: Iterable[*Component*]* | *None* = *None*, *descendants: Iterable[*Component*]* | *None* = *None*, *variants: Iterable[*Component*]* | *None* = *None*, *commits: Iterable[*Commit*]* | *None* = *None*, *patches: Iterable[*Patch*]* | *None* = *None*, *notes: str* | *None* = *None*)

Our internal representation of the *pedigreeType* complex type.

Component pedigree is a way to document complex supply chain scenarios where components are created, distributed, modified, redistributed, combined with other components, etc. Pedigree supports viewing this complex chain from the beginning, the end, or anywhere in the middle. It also provides a way to document variants where the exact relation may not be known.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_pedigreeType

property ancestors: SortedSet['Component']

Describes zero or more components in which a component is derived from. This is commonly used to describe forks from existing projects where the forked version contains a ancestor node containing the original component it was forked from.

For example, Component A is the original component. Component B is the component being used and documented in the BOM. However, Component B contains a pedigree node with a single ancestor documenting Component A - the original component from which Component B is derived from.

Returns:
Set of *Component*

property descendants: SortedSet['Component']

Descendants are the exact opposite of ancestors. This provides a way to document all forks (and their forks) of an original or root component.

Returns:
Set of *Component*

property variants: SortedSet['Component']

Variants describe relations where the relationship between the components are not known. For example, if Component A contains nearly identical code to Component B. They are both related, but it is unclear if one is derived from the other, or if they share a common ancestor.

Returns:
Set of *Component*

property commits: SortedSet[*Commit*]

A list of zero or more commits which provide a trail describing how the component deviates from an ancestor, descendant, or variant.

Returns:
Set of *Commit*

property patches: SortedSet[*Patch*]

A list of zero or more patches describing how the component deviates from an ancestor, descendant, or variant. Patches may be complimentary to commits or may be used in place of commits.

Returns:
Set of *Patch*

property notes: str | None

Notes, observations, and other non-structured commentary describing the components pedigree.

Returns:
str if set else *None*

class cyclonedx.model.component.**Swid**(*, *tag_id: str*, *name: str*, *version: str | None = None*, *tag_version: int | None = None*, *patch: bool | None = None*, *text:* cyclonedx.model.AttachedText *| None = None*, *url:* cyclonedx.model.XsUri *| None = None*)

Our internal representation of the *swidType* complex type.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_swidType

property tag_id: str

Maps to the tagId of a SoftwareIdentity.

Returns:
str

property name: str

Maps to the name of a SoftwareIdentity.

Returns:
str

property version: str | None

Maps to the version of a SoftwareIdentity.

Returns:
str if set else *None*.

property tag_version: int | None

Maps to the tagVersion of a SoftwareIdentity.

Returns:
int if set else *None*

property patch: bool | None

Maps to the patch of a SoftwareIdentity.

Returns:
bool if set else *None*

property text: *cyclonedx.model.AttachedText* | None

Specifies the full content of the SWID tag.

Returns:
AttachedText if set else *None*

property url: *cyclonedx.model.XsUri* | None

The URL to the SWID file.

Returns:
XsUri if set else *None*

class cyclonedx.model.component.**OmniborId**(*id: str*)

Bases: serializable.helpers.BaseHelper

Helper class that allows us to perform validation on data strings that must conform to https://www.iana.org/assignments/uri-schemes/prov/gitoid.

property id: str

classmethod serialize(*o: Any*) → str

classmethod deserialize(*o: Any*) → *OmniborId*

class cyclonedx.model.component.**Swhid**(*id: str*)

Bases: serializable.helpers.BaseHelper

Helper class that allows us to perform validation on data strings that must conform to https://docs.softwareheritage.org/devel/swh-model/persistent-identifiers.html.

property id: str

classmethod serialize(*o: Any*) → str

classmethod deserialize(*o: Any*) → *Swhid*

class cyclonedx.model.component.**Component**(*, *name: str*, *type:* ComponentType = *ComponentType.LIBRARY*, *mime_type: str | None = None*, *bom_ref: str | cyclonedx.model.bom_ref.BomRef | None = None*, *supplier:* cyclonedx.model.contact.OrganizationalEntity *| None = None*, *publisher: str | None = None*, *group: str | None = None*, *version: str | None = None*, *description: str | None = None*, *scope:* ComponentScope *| None = None*, *hashes: Iterable[*cyclonedx.model.HashType*] | None = None*, *licenses: Iterable[cyclonedx.model.license.License] | None = None*, *copyright: str | None = None*, *purl: packageurl.PackageURL | None = None*, *external_references: Iterable[*cyclonedx.model.ExternalReference*] | None = None*, *properties: Iterable[*cyclonedx.model.Property*] | None = None*, *release_notes:* cyclonedx.model.release_note.ReleaseNotes *| None = None*, *cpe: str | None = None*, *swid:* Swid *| None = None*, *pedigree:* Pedigree *| None = None*, *components: Iterable[*Component*] | None = None*, *evidence:* ComponentEvidence *| None = None*, *modified: bool = False*, *manufacturer:* cyclonedx.model.contact.OrganizationalEntity *| None = None*, *authors: Iterable[*cyclonedx.model.contact.OrganizationalContact*] | None = None*, *omnibor_ids: Iterable[*OmniborId*] | None = None*, *swhids: Iterable[*Swhid*] | None = None*, *crypto_properties:* cyclonedx.model.crypto.CryptoProperties *| None = None*, *tags: Iterable[str] | None = None*, *author: str | None = None*)

Bases: *cyclonedx.model.dependency.Dependable*

This is our internal representation of a Component within a Bom.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.3/#type_component

property type: *ComponentType*

Get the type of this Component.

Returns:
Declared type of this Component as *ComponentType*.

property mime_type: str | None

Get any declared mime-type for this Component.

When used on file components, the mime-type can provide additional context about the kind of file being represented such as an image, font, or executable. Some library or framework components may also have an associated mime-type.

Returns:
str if set else *None*

property bom_ref: *cyclonedx.model.bom_ref.BomRef*

An optional identifier which can be used to reference the component elsewhere in the BOM. Every bom-ref MUST be unique within the BOM.

If a value was not provided in the constructor, a UUIDv4 will have been assigned.

Returns:
BomRef

property supplier: *cyclonedx.model.contact.OrganizationalEntity* | None

The organization that supplied the component. The supplier may often be the manufacture, but may also be a distributor or repackager.

Returns:
OrganizationalEntity if set else *None*

property manufacturer: *cyclonedx.model.contact.OrganizationalEntity* | None

The organization that created the component. Manufacturer is common in components created through automated processes. Components created through manual means may have @.*authors* instead.

Returns:
OrganizationalEntity if set else *None*

property authors: SortedSet[*OrganizationalContact*]

The person(s) who created the component. Authors are common in components created through manual processes. Components created through automated means may have @.*manufacturer* instead.

Returns:
Iterable[OrganizationalContact] if set else *None*

property author: str | None

The person(s) or organization(s) that authored the component.

Returns:
str if set else *None*

property publisher: str | None

The person(s) or organization(s) that published the component

Returns:
str if set else *None*

property group: str | None

The grouping name or identifier. This will often be a shortened, single name of the company or project that produced the component, or the source package or domain name. Whitespace and special characters should be avoided.

Examples include: *apache*, *org.apache.commons*, and *apache.org*.

Returns:
str if set else *None*

property name: str

The name of the component.

This will often be a shortened, single name of the component.

Examples: *commons-lang3* and *jquery*.

Returns:
str

property version: str | None

The component version. The version should ideally comply with semantic versioning but is not enforced.

This is NOT optional for CycloneDX Schema Version < 1.4 but was agreed to default to an empty string where a version was not supplied for schema versions < 1.4

Returns:
Declared version of this Component as *str* or *None*

property description: str | None

Get the description of this Component.

Returns:
str if set, else *None*.

property scope: *ComponentScope* | None

Specifies the scope of the component.

If scope is not specified, 'required' scope should be assumed by the consumer of the BOM.

Returns:
ComponentScope or *None*

property hashes: SortedSet[*HashType*]

Optional list of hashes that help specify the integrity of this Component.

Returns:
Set of *HashType*

property licenses: *cyclonedx.model.license.LicenseRepository*

A optional list of statements about how this Component is licensed.

Returns:
Set of *LicenseChoice*

property copyright: str | None

An optional copyright notice informing users of the underlying claims to copyright ownership in a published work.

Returns:
str or *None*

property cpe: str | None

Specifies a well-formed CPE name that conforms to the CPE 2.2 or 2.3 specification. See https://nvd.nist.gov/products/cpe

Returns:
str if set else *None*

property purl: packageurl.PackageURL | None

Specifies the package-url (PURL).

The purl, if specified, must be valid and conform to the specification defined at: https://github.com/package-url/purl-spec

Returns:
PackageURL or *None*

property omnibor_ids: SortedSet[*OmniborId*]

Specifies the OmniBOR Artifact ID. The OmniBOR, if specified, MUST be valid and conform to the specification defined at: https://www.iana.org/assignments/uri-schemes/prov/gitoid

Returns:
Iterable[str] or *None*

property swhids: SortedSet[*Swhid*]

Specifies the Software Heritage persistent identifier (SWHID). The SWHID, if specified, MUST be valid and conform to the specification defined at: https://docs.softwareheritage.org/devel/swh-model/persistent-identifiers.html

Returns:
Iterable[Swhid] if set else *None*

property swid: *Swid* | None

Specifies metadata and content for ISO-IEC 19770-2 Software Identification (SWID) Tags.

Returns:
Swid if set else *None*

property modified: bool

property pedigree: *Pedigree* | None

Component pedigree is a way to document complex supply chain scenarios where components are created, distributed, modified, redistributed, combined with other components, etc.

Returns:
Pedigree if set else *None*

property external_references: SortedSet[*ExternalReference*]

Provides the ability to document external references related to the component or to the project the component describes.

Returns:
Set of *ExternalReference*

property properties: SortedSet[*Property*]

Provides the ability to document properties in a key/value store. This provides flexibility to include data not officially supported in the standard without having to use additional namespaces or create extensions.

Return:
Set of *Property*

property components: SortedSet['Component']

A list of software and hardware components included in the parent component. This is not a dependency tree. It provides a way to specify a hierarchical representation of component assemblies, similar to system -> subsystem -> parts assembly in physical supply chains.

Returns:
Set of *Component*

property evidence: *ComponentEvidence* | None

Provides the ability to document evidence collected through various forms of extraction or analysis.

Returns:
ComponentEvidence if set else *None*

property release_notes: *cyclonedx.model.release_note.ReleaseNotes* | None

Specifies optional release notes.

Returns:
ReleaseNotes or *None*

property crypto_properties: *cyclonedx.model.crypto.CryptoProperties* | None

Cryptographic assets have properties that uniquely define them and that make them actionable for further reasoning. As an example, it makes a difference if one knows the algorithm family (e.g. AES) or the

specific variant or instantiation (e.g. AES-128-GCM). This is because the security level and the algorithm primitive (authenticated encryption) is only defined by the definition of the algorithm variant. The presence of a weak cryptographic algorithm like SHA1 vs. HMAC-SHA1 also makes a difference.

Returns:
CryptoProperties or *None*

property tags: SortedSet[str]

Textual strings that aid in discovery, search, and retrieval of the associated object. Tags often serve as a way to group or categorize similar or related objects by various attributes.

Returns:
Iterable[str]

static for_file(*absolute_file_path: str*, *path_for_bom: str | None*) → *Component*

Helper method to create a Component that represents the provided local file as a Component.

Args:

absolute_file_path:
Absolute path to the file you wish to represent

path_for_bom:
Optionally, if supplied this is the path that will be used to identify the file in the BOM

Returns:
Component representing the supplied file

get_all_nested_components(*include_self: bool = False*) → Set[*Component*]

get_pypi_url() → str

cyclonedx.model.contact

Module Contents

Classes

PostalAddress	This is our internal representation of the *postalAddressType* complex type that can be used in multiple places
OrganizationalContact	This is our internal representation of the *organizationalContact* complex type that can be used in multiple places
OrganizationalEntity	This is our internal representation of the *organizationalEntity* complex type that can be used in multiple places

class cyclonedx.model.contact.**PostalAddress**(*, *bom_ref: str | cyclonedx.model.bom_ref.BomRef | None = None*, *country: str | None = None*, *region: str | None = None*, *locality: str | None = None*, *post_office_box_number: str | None = None*, *postal_code: str | None = None*, *street_address: str | None = None*)

This is our internal representation of the *postalAddressType* complex type that can be used in multiple places within a CycloneDX BOM document.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.6/xml/#type_postalAddressType

property bom_ref: *cyclonedx.model.bom_ref.BomRef* **| None**

An optional identifier which can be used to reference the component elsewhere in the BOM. Every bom-ref MUST be unique within the BOM.

If a value was not provided in the constructor, a UUIDv4 will have been assigned.

Returns:
BomRef

property country: **str | None**

The country name or the two-letter ISO 3166-1 country code.

Returns:
str or *None*

property region: **str | None**

The region or state in the country. For example, Texas.

Returns:
str or *None*

property locality: **str | None**

The locality or city within the country. For example, Austin.

Returns:
str or *None*

property post_office_box_number: **str | None**

The post office box number. For example, 901.

Returns:
str or *None*

property postal_code: **str | None**

The postal code. For example, 78758.

Returns:
str or *None*

property street_address: **str | None**

The street address. For example, 100 Main Street.

Returns:
str or *None*

class cyclonedx.model.contact.**OrganizationalContact**(*, *name: str | None = None*, *phone: str | None = None*, *email: str | None = None*)

This is our internal representation of the *organizationalContact* complex type that can be used in multiple places within a CycloneDX BOM document.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_organizationalContact

property name: **str | None**

Get the name of the contact.

Returns:
str if set else *None*

property email: str | None

Get the email of the contact.

Returns:
str if set else *None*

property phone: str | None

Get the phone of the contact.

Returns:
str if set else *None*

class cyclonedx.model.contact.**OrganizationalEntity**(*, *name: str | None = None*, *urls: Iterable[cyclonedx.model.XsUri] | None = None*, *contacts: Iterable[OrganizationalContact] | None = None*, *address: PostalAddress | None = None*)

This is our internal representation of the *organizationalEntity* complex type that can be used in multiple places within a CycloneDX BOM document.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_organizationalEntity

property name: str | None

Get the name of the organization.

Returns:
str if set else *None*

property address: *PostalAddress* | None

The physical address (location) of the organization.

Returns:
PostalAddress or *None*

property urls: SortedSet[*XsUri*]

Get a list of URLs of the organization. Multiple URLs are allowed.

Returns:
Set of *XsUri*

property contacts: SortedSet[*OrganizationalContact*]

Get a list of contact person at the organization. Multiple contacts are allowed.

Returns:
Set of *OrganizationalContact*

cyclonedx.model.crypto

This set of classes represents cryptoPropertiesType Complex Type in the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

Module Contents

Classes

CryptoAssetType	This is our internal representation of the cryptoPropertiesType.assetType ENUM type within the CycloneDX standard.
CryptoPrimitive	This is our internal representation of the cryptoPropertiesType.algorithmProperties.primitive ENUM type within the
CryptoExecutionEnvironment	This is our internal representation of the cryptoPropertiesType.algorithmProperties.executionEnvironment ENUM type
CryptoImplementationPlatform	This is our internal representation of the cryptoPropertiesType.algorithmProperties.implementationPlatform ENUM type
CryptoCertificationLevel	This is our internal representation of the cryptoPropertiesType.algorithmProperties.certificationLevel ENUM type
CryptoMode	This is our internal representation of the cryptoPropertiesType.algorithmProperties.mode ENUM type
CryptoPadding	This is our internal representation of the cryptoPropertiesType.algorithmProperties.padding ENUM type
CryptoFunction	This is our internal representation of the cryptoPropertiesType.algorithmProperties.cryptoFunctions.cryptoFunction
AlgorithmProperties	This is our internal representation of the cryptoPropertiesType.algorithmProperties ENUM type within the CycloneDX
CertificateProperties	This is our internal representation of the *cryptoPropertiesType.certificateProperties* complex type within
RelatedCryptoMaterialType	This is our internal representation of the cryptoPropertiesType.relatedCryptoMaterialProperties.type ENUM type
RelatedCryptoMaterialState	This is our internal representation of the cryptoPropertiesType.relatedCryptoMaterialProperties.state ENUM type
RelatedCryptoMaterialSecuredBy	This is our internal representation of the *cryptoPropertiesType.relatedCryptoMaterialProperties.securedBy* complex
RelatedCryptoMaterialProperties	This is our internal representation of the *cryptoPropertiesType.relatedCryptoMaterialProperties* complex type
ProtocolPropertiesType	This is our internal representation of the cryptoPropertiesType.protocolProperties.type ENUM type
ProtocolPropertiesCipherSuite	This is our internal representation of the *cryptoPropertiesType.protocolProperties.cipherSuites.cipherSuite*
Ikev2TransformTypes	This is our internal representation of the *cryptoPropertiesType.protocolProperties.ikev2TransformTypes*
ProtocolProperties	This is our internal representation of the *cryptoPropertiesType.protocolProperties* complex type within
CryptoProperties	This is our internal representation of the *cryptoPropertiesType* complex type within CycloneDX standard.

class cyclonedx.model.crypto.**CryptoAssetType**

Bases: str, enum.Enum

This is our internal representation of the cryptoPropertiesType.assetType ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

ALGORITHM = 'algorithm'

CERTIFICATE = 'certificate'

PROTOCOL = 'protocol'

RELATED_CRYPTO_MATERIAL = 'related-crypto-material'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

rfind()

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

rindex()

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

rjust()

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

rpartition()

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

rsplit()

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

`translate()`

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

`upper()`

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.crypto.**CryptoPrimitive**

Bases: str, enum.Enum

This is our internal representation of the cryptoPropertiesType.algorithmProperties.primitive ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

AE = 'ae'

BLOCK_CIPHER = 'block-cipher'

COMBINER = 'combiner'

DRBG = 'drbg'

HASH = 'hash'

KDF = 'kdf'

KEM = 'kem'

KEY_AGREE = 'key-agree'

MAC = 'mac'

PKE = 'pke'

SIGNATURE = 'signature'

STREAM_CIPHER = 'stream-cipher'

XOF = 'xof'

OTHER = 'other'

UNKNOWN = 'unknown'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

`casefold()`

Return a version of the string suitable for caseless comparisons.

`center()`

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

`count()`

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

`encode()`

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

`endswith()`

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

`expandtabs()`

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

`find()`

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`format()`

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

`format_map()`

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

`index()`

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as “def” or “class”.

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

`isupper()`

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

`join()`

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

`ljust()`

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

`lower()`

Return a copy of the string converted to lowercase.

`lstrip()`

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

`partition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

`removeprefix()`

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

`removesuffix()`

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

`replace()`

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

startswith()

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

strip()

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

swapcase()

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.crypto.**CryptoExecutionEnvironment**

Bases: str, enum.Enum

This is our internal representation of the cryptoPropertiesType.algorithmProperties.executionEnvironment ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

HARDWARE = 'hardware'

SOFTWARE_ENCRYPTED_RAM = 'software-encrypted-ram'

SOFTWARE_PLAIN_RAM = 'software-plain-ram'

SOFTWARE_TEE = 'software-tee'

OTHER = 'other'

UNKNOWN = 'unknown'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

`removesuffix()`

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

`replace()`

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

`translate()`

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

`upper()`

Return a copy of the string converted to uppercase.

`zfill()`

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

`name()`

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.crypto.CryptoImplementationPlatform

Bases: str, enum.Enum

This is our internal representation of the cryptoPropertiesType.algorithmProperties.implementationPlatform ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

ARMV7_A = 'armv7-a'

ARMV7_M = 'armv7-m'

ARMV8_A = 'armv8-a'

ARMV8_M = 'armv8-m'

ARMV9_A = 'armv9-a'

ARMV9_M = 'armv9-m'

GENERIC = 'generic'

PPC64 = 'ppc64'

PPC64LE = 'ppc64le'

S390X = 's390x'

X86_32 = 'x86_32'

X86_64 = 'x86_64'

OTHER = 'other'

UNKNOWN = 'unknown'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

`isalpha()`

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

`isascii()`

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

`isdecimal()`

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

`isdigit()`

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

`isidentifier()`

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as “def” or “class”.

`islower()`

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

`isnumeric()`

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

`isprintable()`

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

`isspace()`

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

`istitle()`

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

`isupper()`

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

`join()`

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

`ljust()`

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

`lower()`

Return a copy of the string converted to lowercase.

`lstrip()`

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

`partition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

`removeprefix()`

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

`removesuffix()`

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

`replace()`

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

strip()

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

swapcase()

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.crypto.**CryptoCertificationLevel**

Bases: str, enum.Enum

This is our internal representation of the cryptoPropertiesType.algorithmProperties.certificationLevel ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

---

NONE = 'none'

FIPS140_1_L1 = 'fips140-1-l1'

FIPS140_1_L2 = 'fips140-1-l2'

FIPS140_1_L3 = 'fips140-1-l3'

FIPS140_1_L4 = 'fips140-1-l4'

FIPS140_2_L1 = 'fips140-2-l1'

FIPS140_2_L2 = 'fips140-2-l2'

FIPS140_2_L3 = 'fips140-2-l3'

FIPS140_2_L4 = 'fips140-2-l4'

FIPS140_3_L1 = 'fips140-3-l1'

FIPS140_3_L2 = 'fips140-3-l2'

FIPS140_3_L3 = 'fips140-3-l3'

FIPS140_3_L4 = 'fips140-3-l4'

CC_EAL1 = 'cc-eal1'

CC_EAL1_PLUS = 'cc-eal1+'

CC_EAL2 = 'cc-eal2'

CC_EAL2_PLUS = 'cc-eal2+'

CC_EAL3 = 'cc-eal3'

CC_EAL3_PLUS = 'cc-eal3+'

CC_EAL4 = 'cc-eal4'

CC_EAL4_PLUS = 'cc-eal4+'

CC_EAL5 = 'cc-eal5'

CC_EAL5_PLUS = 'cc-eal5+'

CC_EAL6 = 'cc-eal6'

CC_EAL6_PLUS = 'cc-eal6+'

CC_EAL7 = 'cc-eal7'

CC_EAL7_PLUS = 'cc-eal7+'

OTHER = 'other'

UNKNOWN = 'unknown'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

`isalpha()`

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

`isascii()`

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

`isdecimal()`

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

`isdigit()`

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

`isidentifier()`

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as “def” or “class”.

`islower()`

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

`isnumeric()`

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

`isprintable()`

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

`isspace()`

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

`istitle()`

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

`isupper()`

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

`join()`

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

`ljust()`

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

`lower()`

Return a copy of the string converted to lowercase.

`lstrip()`

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

`partition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

`removeprefix()`

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

`removesuffix()`

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

`replace()`

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

strip()

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

swapcase()

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.crypto.**CryptoMode**

Bases: str, enum.Enum

This is our internal representation of the cryptoPropertiesType.algorithmProperties.mode ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

---

CBC = 'cbc'

CCM = 'ccm'

CFB = 'cfb'

CTR = 'ctr'

ECB = 'ecb'

GCM = 'gcm'

OFB = 'ofb'

OTHER = 'other'

UNKNOWN = 'unknown'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

`isnumeric()`

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

`isprintable()`

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

`isspace()`

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

`istitle()`

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

`isupper()`

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

`join()`

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

`ljust()`

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

`lower()`

Return a copy of the string converted to lowercase.

`lstrip()`

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

`partition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

`removeprefix()`

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

rfind()

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

rindex()

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

rjust()

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

rpartition()

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

rsplit()

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

rstrip()

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

`translate()`

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

`upper()`

Return a copy of the string converted to uppercase.

`zfill()`

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

`name()`

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.crypto.CryptoPadding

Bases: str, enum.Enum

This is our internal representation of the cryptoPropertiesType.algorithmProperties.padding ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

PKCS5 = 'pkcs5'

PKCS7 = 'pkcs7'

PKCS1V15 = 'pkcs1v15'

OAEP = 'oaep'

RAW = 'raw'

OTHER = 'other'

UNKNOWN = 'unknown'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

`lstrip()`

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

`partition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

`removeprefix()`

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

`removesuffix()`

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

`replace()`

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.crypto.**CryptoFunction**

Bases: str, enum.Enum

This is our internal representation of the cryptoPropertiesType.algorithmProperties.cryptoFunctions.cryptoFunction ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

---

DECAPSULATE = 'decapsulate'

DECRYPT = 'decrypt'

DIGEST = 'digest'

ENCAPSULATE = 'encapsulate'

ENCRYPT = 'encrypt'

GENERATE = 'generate'

KEYDERIVE = 'keyderive'

KEYGEN = 'keygen'

SIGN = 'sign'

TAG = 'tag'

VERIFY = 'verify'

OTHER = 'other'

UNKNOWN = 'unknown'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as “def” or “class”.

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

splitlines()

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

startswith()

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

strip()

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

swapcase()

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.crypto.**AlgorithmProperties**(*, *primitive:* CryptoPrimitive | *None* = *None*, *parameter_set_identifier: str* | *None* = *None*, *curve: str* | *None* = *None*, *execution_environment:* CryptoExecutionEnvironment | *None* = *None*, *implementation_platform:* CryptoImplementationPlatform | *None* = *None*, *certification_levels: Iterable[*CryptoCertificationLevel*]* | *None* = *None*, *mode:* CryptoMode | *None* = *None*, *padding:* CryptoPadding | *None* = *None*, *crypto_functions: Iterable[*CryptoFunction*]* | *None* = *None*, *classical_security_level: int* | *None* = *None*, *nist_quantum_security_level: int* | *None* = *None*)

This is our internal representation of the cryptoPropertiesType.algorithmProperties ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

property primitive: *CryptoPrimitive* | **None**

Cryptographic building blocks used in higher-level cryptographic systems and protocols.

Primitives represent different cryptographic routines: deterministic random bit generators (drbg, e.g. CTR_DRBG from NIST SP800-90A-r1), message authentication codes (mac, e.g. HMAC-SHA-256), blockciphers (e.g. AES), streamciphers (e.g. Salsa20), signatures (e.g. ECDSA), hash functions (e.g. SHA-256), public-key encryption schemes (pke, e.g. RSA), extended output functions (xof, e.g. SHAKE256), key derivation functions (e.g. pbkdf2), key agreement algorithms (e.g. ECDH), key encapsulation mechanisms (e.g. ML-KEM), authenticated encryption (ae, e.g. AES-GCM) and the combination of multiple algorithms (combiner, e.g. SP800-56Cr2).

Returns:
CryptoPrimitive or *None*

property parameter_set_identifier: str | None

An identifier for the parameter set of the cryptographic algorithm. Examples: in AES128, '128' identifies the key length in bits, in SHA256, '256' identifies the digest length, '128' in SHAKE128 identifies its maximum security level in bits, and 'SHA2-128s' identifies a parameter set used in SLH-DSA (FIPS205).

Returns:
str or *None*

property curve: str | None

The specific underlying Elliptic Curve (EC) definition employed which is an indicator of the level of security strength, performance and complexity. Absent an authoritative source of curve names, CycloneDX recommends use of curve names as defined at https://neuromancer.sk/std/, the source from which can be found at https://github.com/J08nY/std-curves.

Returns:
str or *None*

property execution_environment: *CryptoExecutionEnvironment* | **None**

The target and execution environment in which the algorithm is implemented in.

Returns:
CryptoExecutionEnvironment or *None*

property implementation_platform: *CryptoImplementationPlatform* | None

The target platform for which the algorithm is implemented. The implementation can be 'generic', running on any platform or for a specific platform.

Returns:
CryptoImplementationPlatform or *None*

property certification_levels: SortedSet[*CryptoCertificationLevel*]

The certification that the implementation of the cryptographic algorithm has received, if any. Certifications include revisions and levels of FIPS 140 or Common Criteria of different Extended Assurance Levels (CC-EAL).

Returns:
Iterable[CryptoCertificationLevel]

property mode: *CryptoMode* | None

The mode of operation in which the cryptographic algorithm (block cipher) is used.

Returns:
CryptoMode or *None*

property padding: *CryptoPadding* | None

The padding scheme that is used for the cryptographic algorithm.

Returns:
CryptoPadding or *None*

property crypto_functions: SortedSet[*CryptoFunction*]

The cryptographic functions implemented by the cryptographic algorithm.

Returns:
Iterable[CryptoFunction]

property classical_security_level: int | None

The classical security level that a cryptographic algorithm provides (in bits).

Returns:
int or *None*

property nist_quantum_security_level: int | None

The NIST security strength category as defined in https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/evaluation-criteria/security-(evaluation-criteria). A value of 0 indicates that none of the categories are met.

Returns:
int or *None*

class cyclonedx.model.crypto.**CertificateProperties**(*, *subject_name: str | None = None*, *issuer_name: str | None = None*, *not_valid_before: datetime.datetime | None = None*, *not_valid_after: datetime.datetime | None = None*, *signature_algorithm_ref: cyclonedx.model.bom_ref.BomRef | None = None*, *subject_public_key_ref: cyclonedx.model.bom_ref.BomRef | None = None*, *certificate_format: str | None = None*, *certificate_extension: str | None = None*)

This is our internal representation of the *cryptoPropertiesType.certificateProperties* complex type within CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

property subject_name: str | None

The subject name for the certificate.

Returns:
str or *None*

property issuer_name: str | None

The issuer name for the certificate.

Returns:
str or *None*

property not_valid_before: datetime.datetime | None

The date and time according to ISO-8601 standard from which the certificate is valid.

Returns:
datetime or *None*

property not_valid_after: datetime.datetime | None

The date and time according to ISO-8601 standard from which the certificate is not valid anymore.

Returns:
datetime or *None*

property signature_algorithm_ref: *cyclonedx.model.bom_ref.BomRef* | None

The bom-ref to signature algorithm used by the certificate.

Returns:
BomRef or *None*

property subject_public_key_ref: *cyclonedx.model.bom_ref.BomRef* | None

The bom-ref to the public key of the subject.

Returns:
BomRef or *None*

property certificate_format: str | None

The format of the certificate. Examples include X.509, PEM, DER, and CVC.

Returns:
str or *None*

property certificate_extension: str | None

The file extension of the certificate. Examples include crt, pem, cer, der, and p12.

Returns:
str or *None*

class cyclonedx.model.crypto.**RelatedCryptoMaterialType**

Bases: str, enum.Enum

This is our internal representation of the cryptoPropertiesType.relatedCryptoMaterialProperties.type ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

ADDITIONAL_DATA = 'additional-data'

CIPHERTEXT = 'ciphertext'

CREDENTIAL = 'credential'

DIGEST = 'digest'

INITIALIZATION_VECTOR = 'initialization-vector'

KEY = 'key'

NONCE = 'nonce'

PASSWORD = 'password'

PRIVATE_KEY = 'private-key'

PUBLIC_KEY = 'public-key'

SALT = 'salt'

SECRET_KEY = 'secret-key'

SEED = 'seed'

SHARED_SECRET = 'shared-secret'

SIGNATURE = 'signature'

TAG = 'tag'

TOKEN = 'token'

OTHER = 'other'

UNKNOWN = 'unknown'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as “def” or “class”.

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

`isupper()`

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

`join()`

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

`ljust()`

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

`lower()`

Return a copy of the string converted to lowercase.

`lstrip()`

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

`partition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

`removeprefix()`

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

`removesuffix()`

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

`replace()`

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

startswith()

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

strip()

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

swapcase()

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.crypto.**RelatedCryptoMaterialState**

Bases: str, enum.Enum

This is our internal representation of the cryptoPropertiesType.relatedCryptoMaterialProperties.state ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

ACTIVE = 'active'

COMPROMISED = 'compromised'

DEACTIVATED = 'deactivated'

DESTROYED = 'destroyed'

PRE_ACTIVATION = 'pre-activation'

SUSPENDED = 'suspended'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

`removesuffix()`

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

`replace()`

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

`translate()`

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

`upper()`

Return a copy of the string converted to uppercase.

`zfill()`

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

`name()`

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.crypto.**RelatedCryptoMaterialSecuredBy**(*, *mechanism: str | None = None*, *algorithm_ref: cyclonedx.model.bom_ref.BomRef | None = None*)

This is our internal representation of the *cryptoPropertiesType.relatedCryptoMaterialProperties.securedBy* complex type within CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

property mechanism: str | None

Specifies the mechanism by which the cryptographic asset is secured by. Examples include HSM, TPM, XGX, Software, and None.

Returns:
str or *None*

property algorithm_ref: *cyclonedx.model.bom_ref.BomRef* | None

The bom-ref to the algorithm.

Returns:
BomRef or *None*

class cyclonedx.model.crypto.**RelatedCryptoMaterialProperties**(*, *type: RelatedCryptoMaterialType | None = None*, *id: str | None = None*, *state: RelatedCryptoMaterialState | None = None*, *algorithm_ref: cyclonedx.model.bom_ref.BomRef | None = None*, *creation_date: datetime.datetime | None = None*, *activation_date: datetime.datetime | None = None*, *update_date: datetime.datetime | None = None*, *expiration_date: datetime.datetime | None = None*, *value: str | None = None*, *size: int | None = None*, *format: str | None = None*, *secured_by: RelatedCryptoMaterialSecuredBy | None = None*)

This is our internal representation of the *cryptoPropertiesType.relatedCryptoMaterialProperties* complex type within CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

property type: *RelatedCryptoMaterialType* | **None**

The type for the related cryptographic material.

Returns

property id: **str | None**

The optional unique identifier for the related cryptographic material.

Returns

property state: *RelatedCryptoMaterialState* | **None**

The key state as defined by NIST SP 800-57.

Returns:
RelatedCryptoMaterialState or *None*

property algorithm_ref: *cyclonedx.model.bom_ref.BomRef* | **None**

The bom-ref to the algorithm used to generate the related cryptographic material.

Returns:
BomRef or *None*

property creation_date: **datetime.datetime | None**

The date and time (timestamp) when the related cryptographic material was created.

Returns:
datetime or *None*

property activation_date: **datetime.datetime | None**

The date and time (timestamp) when the related cryptographic material was activated.

Returns:
datetime or *None*

property update_date: **datetime.datetime | None**

The date and time (timestamp) when the related cryptographic material was updated.

Returns:
datetime or *None*

property expiration_date: **datetime.datetime | None**

The date and time (timestamp) when the related cryptographic material expires.

Returns:
datetime or *None*

property value: **str | None**

The associated value of the cryptographic material.

Returns:
str or *None*

property size: **int | None**

The size of the cryptographic asset (in bits).

Returns:
int or *None*

property format: str | None

The format of the related cryptographic material (e.g. P8, PEM, DER).

Returns:
str or *None*

property secured_by: *RelatedCryptoMaterialSecuredBy* | None

The mechanism by which the cryptographic asset is secured by.

Returns:
RelatedCryptoMaterialSecuredBy or *None*

class cyclonedx.model.crypto.**ProtocolPropertiesType**

Bases: str, enum.Enum

This is our internal representation of the cryptoPropertiesType.protocolProperties.type ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

IKE = 'ike'

IPSEC = 'ipsec'

SSH = 'ssh'

SSTP = 'sstp'

TLS = 'tls'

WPA = 'wpa'

OTHER = 'other'

UNKNOWN = 'unknown'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

rfind()

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

rindex()

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

rjust()

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

`translate()`

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

`upper()`

Return a copy of the string converted to uppercase.

`zfill()`

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

`name()`

The name of the Enum member.

`value()`

The value of the Enum member.

`class cyclonedx.model.crypto.ProtocolPropertiesCipherSuite(*, name: str | None = None, algorithms: Iterable[cyclonedx.model.bom_ref.BomRef] | None = None, identifiers: Iterable[str] | None = None)`

This is our internal representation of the *cryptoPropertiesType.protocolProperties.cipherSuites.cipherSuite* complex type within CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

---

`property name: str | None`

A common name for the cipher suite. For example: TLS_DHE_RSA_WITH_AES_128_CCM.

Returns:
str or *None*

`property algorithms: SortedSet[BomRef]`

A list BomRefs to algorithms related to the cipher suite.

Returns:
Iterable[BomRef] or *None*

property identifiers: SortedSet[str]

A list of common identifiers for the cipher suite. Examples include 0xC0 and 0x9E.

Returns:
Iterable[str] or *None*

class cyclonedx.model.crypto.**Ikev2TransformTypes**(*, *encr: Iterable[cyclonedx.model.bom_ref.BomRef] | None = None*, *prf: Iterable[cyclonedx.model.bom_ref.BomRef] | None = None*, *integ: Iterable[cyclonedx.model.bom_ref.BomRef] | None = None*, *ke: Iterable[cyclonedx.model.bom_ref.BomRef] | None = None*, *esn: bool | None = None*, *auth: Iterable[cyclonedx.model.bom_ref.BomRef] | None = None*)

This is our internal representation of the *cryptoPropertiesType.protocolProperties.ikev2TransformTypes* complex type within CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

---

property encr: SortedSet[*BomRef*]

Transform Type 1: encryption algorithms.

Returns:
Iterable[BomRef] or *None*

property prf: SortedSet[*BomRef*]

Transform Type 2: pseudorandom functions.

Returns:
Iterable[BomRef] or *None*

property integ: SortedSet[*BomRef*]

Transform Type 3: integrity algorithms.

Returns:
Iterable[BomRef] or *None*

property ke: SortedSet[*BomRef*]

Transform Type 4: Key Exchange Method (KE) per RFC9370, formerly called Diffie-Hellman Group (D-H).

Returns:
Iterable[BomRef] or *None*

property esn: bool | None

Specifies if an Extended Sequence Number (ESN) is used.

Returns:
bool or *None*

property auth: SortedSet[*BomRef*]

IKEv2 Authentication method.

Returns:
Iterable[BomRef] or *None*

class cyclonedx.model.crypto.**ProtocolProperties**(*, *type:* ProtocolPropertiesType | *None* = *None*, *version: str* | *None* = *None*, *cipher_suites:* *Iterable[*ProtocolPropertiesCipherSuite*]* | *None* = *None*, *ikev2_transform_types:* Ikev2TransformTypes | *None* = *None*)

This is our internal representation of the *cryptoPropertiesType.protocolProperties* complex type within CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

property type: *ProtocolPropertiesType* | None

The concrete protocol type.

Returns:
ProtocolPropertiesType or *None*

property version: str | None

The version of the protocol. Examples include 1.0, 1.2, and 1.99.

Returns:
str or *None*

property cipher_suites: SortedSet[*ProtocolPropertiesCipherSuite*]

A list of cipher suites related to the protocol.

Returns:
Iterable[ProtocolPropertiesCipherSuite]

property ikev2_transform_types: *Ikev2TransformTypes* | None

The IKEv2 transform types supported (types 1-4), defined in RFC7296 section 3.3.2, and additional properties.

Returns:
Ikev2TransformTypes or *None*

class cyclonedx.model.crypto.**CryptoProperties**(*, *asset_type:* CryptoAssetType | *None* = *None*, *algorithm_properties:* AlgorithmProperties | *None* = *None*, *certificate_properties:* CertificateProperties | *None* = *None*, *related_crypto_material_properties:* RelatedCryptoMaterialProperties | *None* = *None*, *protocol_properties:* ProtocolProperties | *None* = *None*, *oid: str* | *None* = *None*)

This is our internal representation of the *cryptoPropertiesType* complex type within CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_cryptoPropertiesType

property asset_type: *CryptoAssetType* | None

Cryptographic assets occur in several forms. Algorithms and protocols are most commonly implemented in specialized cryptographic libraries. They may however also be 'hardcoded' in software components. Certificates and related cryptographic material like keys, tokens, secrets or passwords are other cryptographic assets to be modelled.

Returns:
CryptoAssetType

property algorithm_properties: *AlgorithmProperties* | None

Additional properties specific to a cryptographic algorithm.

Returns:
AlgorithmProperties or *None*

property certificate_properties: *CertificateProperties* | None

Properties for cryptographic assets of asset type 'certificate'.

Returns:
CertificateProperties or *None*

property related_crypto_material_properties: *RelatedCryptoMaterialProperties* | None

Properties for cryptographic assets of asset type 'relatedCryptoMaterial'.

Returns:
RelatedCryptoMaterialProperties or *None*

property protocol_properties: *ProtocolProperties* | None

Properties specific to cryptographic assets of type: 'protocol'.

Returns:
ProtocolProperties or *None*

property oid: str | None

The object identifier (OID) of the cryptographic asset.

Returns:
str or *None*

cyclonedx.model.dependency

Module Contents

Classes

Dependency	Models a Dependency within a BOM.
Dependable	Dependable objects can be part of the Dependency Graph

class cyclonedx.model.dependency.**Dependency**(*ref:* cyclonedx.model.bom_ref.BomRef, *dependencies: Iterable[*Dependency*]* | *None* = *None*)

Models a Dependency within a BOM.

Note: See https://cyclonedx.org/docs/1.4/xml/#type_dependencyType

property ref: *cyclonedx.model.bom_ref.BomRef*

property dependencies: SortedSet[*Dependency*]

dependencies_as_bom_refs() → Set[*cyclonedx.model.bom_ref.BomRef*]

class cyclonedx.model.dependency.**Dependable**

Bases: abc.ABC

Dependable objects can be part of the Dependency Graph

abstract property bom_ref: *cyclonedx.model.bom_ref.BomRef*

cyclonedx.model.impact_analysis

This set of classes represents the data about Impact Analysis.

Impact Analysis is new for CycloneDX schema version 1.

Note: See the CycloneDX Schema extension definition https://cyclonedx.org/docs/1.4

Module Contents

Classes

ImpactAnalysisAffectedStatus	Enum object that defines the permissible impact analysis affected states.
ImpactAnalysisJustification	Enum object that defines the rationale of why the impact analysis state was asserted.
ImpactAnalysisResponse	Enum object that defines the valid rationales as to why the impact analysis state was asserted.
ImpactAnalysisState	Enum object that defines the permissible impact analysis states.

class cyclonedx.model.impact_analysis.**ImpactAnalysisAffectedStatus**

Bases: str, enum.Enum

Enum object that defines the permissible impact analysis affected states.

The vulnerability status of a given version or range of versions of a product.

The statuses 'affected' and 'unaffected' indicate that the version is affected or unaffected by the vulnerability.

The status 'unknown' indicates that it is unknown or unspecified whether the given version is affected. There can be many reasons for an 'unknown' status, including that an investigation has not been undertaken or that a vendor has not disclosed the status.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/#type_impactAnalysisAffectedStatusType

---

`AFFECTED = 'affected'`

`UNAFFECTED = 'unaffected'`

`UNKNOWN = 'unknown'`

`capitalize()`

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

`casefold()`

Return a version of the string suitable for caseless comparisons.

`center()`

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

`count()`

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

`encode()`

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

`endswith()`

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

`expandtabs()`

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

`find()`

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

`isnumeric()`

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

`isprintable()`

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

`isspace()`

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

`istitle()`

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

`isupper()`

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

`join()`

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

`ljust()`

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

`lower()`

Return a copy of the string converted to lowercase.

`lstrip()`

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

`partition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

`removeprefix()`

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

rfind()

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

rindex()

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

rjust()

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

rpartition()

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

rsplit()

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

rstrip()

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

`translate()`

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

`upper()`

Return a copy of the string converted to uppercase.

`zfill()`

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

`name()`

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.impact_analysis.**ImpactAnalysisJustification**

Bases: str, enum.Enum

Enum object that defines the rationale of why the impact analysis state was asserted.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/#type_impactAnalysisJustificationType

---

CODE_NOT_PRESENT = 'code_not_present'

CODE_NOT_REACHABLE = 'code_not_reachable'

PROTECTED_AT_PERIMITER = 'protected_at_perimeter'

PROTECTED_AT_RUNTIME = 'protected_at_runtime'

PROTECTED_BY_COMPILER = 'protected_by_compiler'

PROTECTED_BY_MITIGATING_CONTROL = 'protected_by_mitigating_control'

REQUIRES_CONFIGURATION = 'requires_configuration'

REQUIRES_DEPENDENCY = 'requires_dependency'

REQUIRES_ENVIRONMENT = 'requires_environment'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

`lstrip()`

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

`partition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

`removeprefix()`

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

`removesuffix()`

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

`replace()`

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.impact_analysis.**ImpactAnalysisResponse**

Bases: str, enum.Enum

Enum object that defines the valid rationales as to why the impact analysis state was asserted.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/#type_impactAnalysisResponsesType

---

CAN_NOT_FIX = 'can_not_fix'

ROLLBACK = 'rollback'

UPDATE = 'update'

WILL_NOT_FIX = 'will_not_fix'

WORKAROUND_AVAILABLE = 'workaround_available'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

rfind()

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

rindex()

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

rjust()

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.impact_analysis.**ImpactAnalysisState**

Bases: str, enum.Enum

Enum object that defines the permissible impact analysis states.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/#type_impactAnalysisStateType

---

RESOLVED = 'resolved'

RESOLVED_WITH_PEDIGREE = 'resolved_with_pedigree'

EXPLOITABLE = 'exploitable'

IN_TRIAGE = 'in_triage'

FALSE_POSITIVE = 'false_positive'

NOT_AFFECTED = 'not_affected'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as “def” or “class”.

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()
: Return True if the string is an uppercase string, False otherwise.

  A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()
: Concatenate any number of strings.

  The string whose method is called is inserted in between each given string. The result is returned as a new string.

  Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()
: Return a left-justified string of length width.

  Padding is done using the specified fill character (default is a space).

lower()
: Return a copy of the string converted to lowercase.

lstrip()
: Return a copy of the string with leading whitespace removed.

  If chars is given and not None, remove characters in chars instead.

partition()
: Partition the string into three parts using the given separator.

  This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

  If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()
: Return a str with the given prefix string removed if present.

  If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()
: Return a str with the given suffix string removed if present.

  If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()
: Return a copy with all occurrences of substring old replaced by new.

  count
  : Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

  If the optional argument count is given, only the first count occurrences are replaced.

rfind()
: S.rfind(sub[, start[, end]]) -> int

  Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

  Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

startswith()

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

strip()

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

swapcase()

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

cyclonedx.model.issue

Module Contents

Classes

IssueClassification	This is our internal representation of the enum *issueClassification*.
IssueTypeSource	This is our internal representation ofa source within the IssueType complex type that can be used in multiple
IssueType	This is our internal representation of an IssueType complex type that can be used in multiple places within

class cyclonedx.model.issue.**IssueClassification**

Bases: str, enum.Enum

This is our internal representation of the enum *issueClassification*.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_issueClassification

DEFECT = 'defect'

ENHANCEMENT = 'enhancement'

SECURITY = 'security'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

`removesuffix()`

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

`replace()`

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

`translate()`

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

`upper()`

Return a copy of the string converted to uppercase.

`zfill()`

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

`name()`

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.issue.**IssueTypeSource**(*, *name: str | None = None*, *url:* cyclonedx.model.XsUri | *None = None*)

This is our internal representation ofa source within the IssueType complex type that can be used in multiple places within a CycloneDX BOM document.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_issueType

property name: str | None

The name of the source. For example "National Vulnerability Database", "NVD", and "Apache".

Returns:
str if set else *None*

property url: *cyclonedx.model.XsUri* | None

Optional url of the issue documentation as provided by the source.

Returns:
XsUri if set else *None*

class cyclonedx.model.issue.**IssueType**(*, *type:* IssueClassification, *id: str | None = None*, *name: str | None = None*, *description: str | None = None*, *source:* IssueTypeSource | *None = None*, *references:* *Iterable[*cyclonedx.model.XsUri*]* | *None = None*)

This is our internal representation of an IssueType complex type that can be used in multiple places within a CycloneDX BOM document.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_issueType

property type: *IssueClassification*

Specifies the type of issue.

Returns:
IssueClassification

property id: str | None

The identifier of the issue assigned by the source of the issue.

Returns:
str if set else *None*

property name: str | None

The name of the issue.

Returns:
str if set else *None*

property description: str | None

A description of the issue.

Returns:
str if set else *None*

property source: *IssueTypeSource* | **None**

The source of this issue.

Returns:
IssueTypeSource if set else *None*

property references: **SortedSet[*XsUri*]**

Any reference URLs related to this issue.

Returns:
Set of *XsUri*

cyclonedx.model.license

License related things

Module Contents

Classes

LicenseAcknowledgement	This is our internal representation of the *type_licenseAcknowledgementEnumerationType* ENUM type
DisjunctiveLicense	This is our internal representation of *licenseType* complex type that can be used in multiple places within
LicenseExpression	This is our internal representation of *licenseType*'s expression type that can be used in multiple places within
LicenseRepository	Collection of *License*.

Attributes

LicenseExpressionAcknowledgement	Deprecated alias for *LicenseAcknowledgement*
License	TypeAlias for a union of supported license models.

class cyclonedx.model.license.**LicenseAcknowledgement**

Bases: `str`, `enum.Enum`

This is our internal representation of the *type_licenseAcknowledgementEnumerationType* ENUM type within the CycloneDX standard.

Note: Introduced in CycloneDX v1.6

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.6/#type_licenseAcknowledgementEnumerationType

CONCLUDED = 'concluded'

DECLARED = 'declared'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as “def” or “class”.

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

splitlines()

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

startswith()

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

strip()

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

swapcase()

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

cyclonedx.model.license.**LicenseExpressionAcknowledgement**

Deprecated alias for *LicenseAcknowledgement*

class cyclonedx.model.license.**DisjunctiveLicense**(*, *id: str | None = None*, *name: str | None = None*, *text:* cyclonedx.model.AttachedText | *None = None*, *url:* cyclonedx.model.XsUri | *None = None*, *acknowledgement:* LicenseAcknowledgement | *None = None*)

This is our internal representation of *licenseType* complex type that can be used in multiple places within a CycloneDX BOM document.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/json/#components_items_licenses

property id: str | None

A SPDX license ID.

Note: See the list of expected values: https://cyclonedx.org/docs/1.4/json/#components_items_licenses_items_license_id

Returns:
str or *None*

property name: str | None

If SPDX does not define the license used, this field may be used to provide the license name.

Returns:
str or *None*

property text: *cyclonedx.model.AttachedText* | None

Specifies the optional full text of the attachment

Returns:
AttachedText else *None*

property url: *cyclonedx.model.XsUri* | None

The URL to the attachment file. If the attachment is a license or BOM, an externalReference should also be specified for completeness.

Returns:
XsUri or *None*

property acknowledgement: *LicenseAcknowledgement* | None

Declared licenses and concluded licenses represent two different stages in the licensing process within software development.

Declared licenses refer to the initial intention of the software authors regarding the licensing terms under which their code is released. On the other hand, concluded licenses are the result of a comprehensive analysis of the project's codebase to identify and confirm the actual licenses of the components used, which may differ from the initially declared licenses. While declared licenses provide an upfront indication of the licensing intentions, concluded licenses offer a more thorough understanding of the actual licensing within a project, facilitating proper compliance and risk management. Observed licenses are defined in evidence.licenses. Observed licenses form the evidence necessary to substantiate a concluded license.

Returns:
LicenseAcknowledgement or *None*

class cyclonedx.model.license.**LicenseExpression**(*value: str*, *acknowledgement:* LicenseAcknowledgement | *None* = *None*)

This is our internal representation of *licenseType*'s expression type that can be used in multiple places within a CycloneDX BOM document.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/json/#components_items_licenses_items_expression

property value: str

Value of this LicenseExpression.

Returns:
str

property acknowledgement: *LicenseAcknowledgement* | None

Declared licenses and concluded licenses represent two different stages in the licensing process within software development.

Declared licenses refer to the initial intention of the software authors regarding the licensing terms under which their code is released. On the other hand, concluded licenses are the result of a comprehensive analysis of the project's codebase to identify and confirm the actual licenses of the components used, which may differ from the initially declared licenses. While declared licenses provide an upfront indication of the licensing intentions, concluded licenses offer a more thorough understanding of the actual licensing within a project, facilitating proper compliance and risk management. Observed licenses are defined in evidence.licenses. Observed licenses form the evidence necessary to substantiate a concluded license.

Returns:
LicenseAcknowledgement or *None*

cyclonedx.model.license.**License**

TypeAlias for a union of supported license models.

- *LicenseExpression*
- *DisjunctiveLicense*

class cyclonedx.model.license.**LicenseRepository**

Bases: sortedcontainers.SortedSet[*License*]

Collection of *License*.

This is a *set*, not a *list*. Order MUST NOT matter here. If you wanted a certain order, then you should also express whether the items are concat by *AND* or *OR*. If you wanted to do so, you should use *LicenseExpression*.

As a model, this MUST accept multiple *LicenseExpression* along with multiple *DisjunctiveLicense*, as this was an accepted in CycloneDX JSON before v1.5. So for modeling purposes, this is supported. Denormalizers/deserializers will be thankful. The normalization/serialization process SHOULD take care of these facts and do what is needed.

cyclonedx.model.release_note

Module Contents

Classes

ReleaseNotes	This is our internal representation of a *releaseNotesType* for a Component in a BOM.

class cyclonedx.model.release_note.**ReleaseNotes**(*, *type: str*, *title: str | None = None*, *featured_image: cyclonedx.model.XsUri | None = None*, *social_image: cyclonedx.model.XsUri | None = None*, *description: str | None = None*, *timestamp: datetime.datetime | None = None*, *aliases: Iterable[str] | None = None*, *tags: Iterable[str] | None = None*, *resolves: Iterable[cyclonedx.model.issue.IssueType] | None = None*, *notes: Iterable[cyclonedx.model.Note] | None = None*, *properties: Iterable[cyclonedx.model.Property] | None = None*)

This is our internal representation of a *releaseNotesType* for a Component in a BOM.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/#type_releaseNotesType

---

property type: str

The software versioning type.

It is **RECOMMENDED** that the release type use one of 'major', 'minor', 'patch', 'pre-release', or 'internal'.

Representing all possible software release types is not practical, so standardizing on the recommended values, whenever possible, is strongly encouraged.

- **major** = A major release may contain significant changes or may introduce breaking changes.
- **minor = A minor release, also known as an update, may contain a smaller number of changes than major**
  releases.
- **patch** = Patch releases are typically unplanned and may resolve defects or important security issues.
- **pre-release = A pre-release may include alpha, beta, or release candidates and typically have limited**
  support. They provide the ability to preview a release prior to its general availability.
- **internal = Internal releases are not for public consumption and are intended to be used exclusively by the**
  project or manufacturer that produced it.

property title: str | None

The title of the release.

property featured_image: *cyclonedx.model.XsUri* | None

The URL to an image that may be prominently displayed with the release note.

property social_image: *cyclonedx.model.XsUri* | None

The URL to an image that may be used in messaging on social media platforms.

property description: str | None

A short description of the release.

property timestamp: datetime.datetime | None

The date and time (timestamp) when the release note was created.

property aliases: SortedSet[str]

One or more alternate names the release may be referred to. This may include unofficial terms used by development and marketing teams (e.g. code names).

Returns:
Set of *str*

property tags: SortedSet[str]

One or more tags that may aid in search or retrieval of the release note.

Returns:
Set of *str*

property resolves: SortedSet[*IssueType*]

A collection of issues that have been resolved.

Returns:
Set of *IssueType*

property notes: SortedSet[*Note*]

Zero or more release notes containing the locale and content. Multiple note elements may be specified to support release notes in a wide variety of languages.

Returns:
Set of *Note*

property properties: SortedSet[*Property*]

Provides the ability to document properties in a name-value store. This provides flexibility to include data not officially supported in the standard without having to use additional namespaces or create extensions. Unlike key-value stores, properties support duplicate names, each potentially having different values.

Returns:
Set of *Property*

cyclonedx.model.service

This set of classes represents the data that is possible about known Services.

Note: See the CycloneDX Schema extension definition https://cyclonedx.org/docs/1.4/xml/#type_servicesType

Module Contents

Classes

Service	Class that models the *service* complex type in the CycloneDX schema.

class cyclonedx.model.service.**Service**(*, *name: str*, *bom_ref: str | cyclonedx.model.bom_ref.BomRef | None = None*, *provider: cyclonedx.model.contact.OrganizationalEntity | None = None*, *group: str | None = None*, *version: str | None = None*, *description: str | None = None*, *endpoints: Iterable[cyclonedx.model.XsUri] | None = None*, *authenticated: bool | None = None*, *x_trust_boundary: bool | None = None*, *data: Iterable[cyclonedx.model.DataClassification] | None = None*, *licenses: Iterable[cyclonedx.model.license.License] | None = None*, *external_references: Iterable[cyclonedx.model.ExternalReference] | None = None*, *properties: Iterable[cyclonedx.model.Property] | None = None*, *services: Iterable[Service] | None = None*, *release_notes: cyclonedx.model.release_note.ReleaseNotes | None = None*)

Bases: *cyclonedx.model.dependency.Dependable*

Class that models the *service* complex type in the CycloneDX schema.

Note: See the CycloneDX schema: https://cyclonedx.org/docs/1.4/xml/#type_service

property bom_ref: *cyclonedx.model.bom_ref.BomRef*

An optional identifier which can be used to reference the service elsewhere in the BOM. Uniqueness is enforced within all elements and children of the root-level bom element.

If a value was not provided in the constructor, a UUIDv4 will have been assigned.

Returns:
BomRef unique identifier for this Service

property provider: *cyclonedx.model.contact.OrganizationalEntity* **| None**

Get the organization that provides the service.

Returns:
OrganizationalEntity if set else *None*

property group: str | None

The grouping name, namespace, or identifier. This will often be a shortened, single name of the company or project that produced the service or domain name. Whitespace and special characters should be avoided.

Returns:
str if provided else *None*

property name: str

The name of the service. This will often be a shortened, single name of the service.

Returns:
str

property version: str | None

The service version.

Returns:
str if set else *None*

property description: str | None

Specifies a description for the service.

Returns:
str if set else *None*

property endpoints: SortedSet[*XsUri*]

A list of endpoints URI's this service provides.

Returns:
Set of *XsUri*

property authenticated: bool | None

A boolean value indicating if the service requires authentication. A value of true indicates the service requires authentication prior to use.

A value of false indicates the service does not require authentication.

Returns:
bool if set else *None*

property x_trust_boundary: bool | None

A boolean value indicating if use of the service crosses a trust zone or boundary. A value of true indicates that by using the service, a trust boundary is crossed.

A value of false indicates that by using the service, a trust boundary is not crossed.

Returns:
bool if set else *None*

property data: SortedSet[*DataClassification*]

Specifies the data classification.

Returns:
Set of *DataClassification*

property licenses: *cyclonedx.model.license.LicenseRepository*

A optional list of statements about how this Service is licensed.

Returns:
Set of *LicenseChoice*

property external_references: SortedSet[*ExternalReference*]

Provides the ability to document external references related to the Service.

Returns:
Set of *ExternalReference*

property properties: SortedSet[*Property*]

Provides the ability to document properties in a key/value store. This provides flexibility to include data not officially supported in the standard without having to use additional namespaces or create extensions.

Return:
Set of *Property*

property services: SortedSet['Service']

A list of services included or deployed behind the parent service.

This is not a dependency tree.

It provides a way to specify a hierarchical representation of service assemblies.

Returns:
Set of *Service*

property release_notes: *cyclonedx.model.release_note.ReleaseNotes* | **None**

Specifies optional release notes.

Returns:
ReleaseNotes or *None*

cyclonedx.model.vulnerability

This set of classes represents the data that is possible about known Vulnerabilities.

Prior to CycloneDX schema version 1.4, vulnerabilities were possible in XML versions ONLY of the standard through a schema extension: https://cyclonedx.org/ext/vulnerability.

Since CycloneDX schema version 1.4, this has become part of the core schema.

Note: See the CycloneDX Schema extension definition https://cyclonedx.org/docs/1.4/#type_vulnerabilitiesType

Module Contents

Classes

BomTargetVersionRange	Class that represents either a version or version range and its affected status.
BomTarget	Class that represents referencing a Component or Service in a BOM.
VulnerabilityAnalysis	Class that models the *analysis* sub-element of the *vulnerabilityType* complex type.
VulnerabilityAdvisory	Class that models the *advisoryType* complex type.
VulnerabilitySource	Class that models the *vulnerabilitySourceType* complex type.
VulnerabilityReference	Class that models the nested *reference* within the *vulnerabilityType* complex type.
VulnerabilityScoreSource	Enum object that defines the permissible source types for a Vulnerability's score.
VulnerabilitySeverity	Class that defines the permissible severities for a Vulnerability.
VulnerabilityRating	Class that models the *ratingType* complex element CycloneDX core schema.
VulnerabilityCredits	Class that models the *credits* of *vulnerabilityType* complex type in the CycloneDX schema (version >= 1.4).
Vulnerability	Class that models the *vulnerabilityType* complex type in the CycloneDX schema (version >= 1.4).

class cyclonedx.model.vulnerability.**BomTargetVersionRange**(*, *version: str* | *None* = *None*, *range: str* | *None* = *None*, *status:* cyclonedx.model.impact_analysis.ImpactAnalysisAffectedSta | *None* = *None*)

Class that represents either a version or version range and its affected status.

version and *version_range* are mutually exclusive.

Note: See the CycloneDX schema: https://cyclonedx.org/docs/1.4/#type_vulnerabilityType

property version: str | None

A single version of a component or service.

property range: str | None

A version range specified in Package URL Version Range syntax (vers) which is defined at https://github.com/package-url/purl-spec/VERSION-RANGE-SPEC.rst

Note: The VERSION-RANGE-SPEC from Package URL is not a formalised standard at the time of writing and this no validation of conformance with this draft standard is performed.

property status: *cyclonedx.model.impact_analysis.ImpactAnalysisAffectedStatus* | None

The vulnerability status for the version or range of versions.

class cyclonedx.model.vulnerability.**BomTarget**(*, *ref: str*, *versions: Iterable[*BomTargetVersionRange*]* | *None* = *None*)

Class that represents referencing a Component or Service in a BOM.

Aims to represent the sub-element *target* of the complex type *vulnerabilityType*.

You can either create a *cyclonedx.model.bom.Bom* yourself programmatically, or generate a *cyclonedx.model.bom.Bom* from a *cyclonedx.parser.BaseParser* implementation.

Note: See the CycloneDX schema: https://cyclonedx.org/docs/1.4/#type_vulnerabilityType

property ref: str

Reference to a component or service by the objects *bom-ref*.

property versions: SortedSet[*BomTargetVersionRange*]

Zero or more individual versions or range of versions.

Returns:
Set of *BomTargetVersionRange*

class cyclonedx.model.vulnerability.**VulnerabilityAnalysis**(*, *state:* cyclonedx.model.impact_analysis.ImpactAnalysisState | *None* = *None*, *justification:* cyclonedx.model.impact_analysis.ImpactAnalysisJustification | *None* = *None*, *responses: Iterable[*cyclonedx.model.impact_analysis.ImpactAnalysisResp | *None* = *None*, *detail: str* | *None* = *None*)

Class that models the *analysis* sub-element of the *vulnerabilityType* complex type.

Note: See the CycloneDX schema: https://cyclonedx.org/docs/1.4/#type_vulnerabilityType

property state: *cyclonedx.model.impact_analysis.ImpactAnalysisState* | None

The declared current state of an occurrence of a vulnerability, after automated or manual analysis.

Returns:
ImpactAnalysisState if set else *None*

property justification: *cyclonedx.model.impact_analysis.ImpactAnalysisJustification* **| None**

The rationale of why the impact analysis state was asserted.

Returns:
ImpactAnalysisJustification if set else *None*

property responses: **SortedSet[***ImpactAnalysisResponse***]**

A list of responses to the vulnerability by the manufacturer, supplier, or project responsible for the affected component or service. More than one response is allowed. Responses are strongly encouraged for vulnerabilities where the analysis state is exploitable.

Returns:
Set of *ImpactAnalysisResponse*

property detail: **str | None**

A detailed description of the impact including methods used during assessment. If a vulnerability is not exploitable, this field should include specific details on why the component or service is not impacted by this vulnerability.

Returns:
str if set else *None*

class cyclonedx.model.vulnerability.**VulnerabilityAdvisory**(*, *url:* cyclonedx.model.XsUri, *title: str | None = None*)

Class that models the *advisoryType* complex type.

Note: See the CycloneDX schema: https://cyclonedx.org/docs/1.4/#type_advisoryType

property title: **str | None**

The title of this advisory.

property url: *cyclonedx.model.XsUri*

The url of this advisory.

class cyclonedx.model.vulnerability.**VulnerabilitySource**(*, *name: str | None = None*, *url:* cyclonedx.model.XsUri *| None = None*)

Class that models the *vulnerabilitySourceType* complex type.

This type is used for multiple purposes in the CycloneDX schema.

Note: See the CycloneDX schema: https://cyclonedx.org/docs/1.4/#type_vulnerabilitySourceType

property name: **str | None**

Name of this Source.

property url: *cyclonedx.model.XsUri* **| None**

The url of this Source.

class cyclonedx.model.vulnerability.**VulnerabilityReference**(*, *id: str | None = None*, *source:* VulnerabilitySource *| None = None*)

Class that models the nested *reference* within the *vulnerabilityType* complex type.

Vulnerabilities may benefit from pointers to vulnerabilities that are the equivalent of the vulnerability specified. Often times, the same vulnerability may exist in multiple sources of vulnerability intelligence, but have different identifiers. These references provide a way to correlate vulnerabilities across multiple sources of vulnerability intelligence.

Note: See the CycloneDX schema: https://cyclonedx.org/docs/1.4/#type_vulnerabilityType

---

property id: str | None

The identifier that uniquely identifies the vulnerability in the associated Source. For example: CVE-2021-39182.

property source: *VulnerabilitySource* | None

The source that published the vulnerability.

class cyclonedx.model.vulnerability.**VulnerabilityScoreSource**

Bases: str, enum.Enum

Enum object that defines the permissible source types for a Vulnerability's score.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/#type_scoreSourceType

---

Note:

No explicit carry-over from the former schema extension:
https://github.com/CycloneDX/specification/blob/master/schema/ext/vulnerability-1.0.xsd

---

CVSS_V2 = 'CVSSv2'

CVSS_V3 = 'CVSSv3'

CVSS_V3_1 = 'CVSSv31'

CVSS_V4 = 'CVSSv4'

OWASP = 'OWASP'

SSVC = 'SSVC'

OTHER = 'other'

static get_from_vector(*vector: str*) → *VulnerabilityScoreSource*

Attempt to derive the correct SourceType from an attack vector.

For example, often attack vector strings are prefixed with the scheme in question - such that __CVSS:3.0/AV:L/AC:L/PR:N/UI:R/S:C/C:L/I:N/**A:N**__ would be the vector __AV:L/AC:L/PR:N/UI:R/S:C/C:L/I:N/**A:N**__ under the __CVSS **3**__ scheme.

Returns:

Always returns an instance of *VulnerabilityScoreSource*. *VulnerabilityScoreSource.OTHER* is returned if the scheme is not obvious or known to us.

`get_localised_vector`(*vector: str*) → str

This method will remove any Source Scheme type from the supplied vector, returning just the vector.

Note: Currently supports CVSS 3.x, CVSS 2.x and OWASP schemes.

Returns:
The vector without any scheme prefix as a *str*.

`get_value_pre_1_4`() → str

Some of the enum values changed in 1.4 of the CycloneDX spec. This method allows us to backport some of the changes for pre-1.4.

Returns:
str

`capitalize`()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

`casefold`()

Return a version of the string suitable for caseless comparisons.

`center`()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

`count`()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

`encode`()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

`endswith`()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

`expandtabs`()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

rfind()

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

rindex()

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

rjust()

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

rpartition()

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

rsplit()

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

`translate()`

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

`upper()`

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.vulnerability.**VulnerabilitySeverity**

Bases: str, enum.Enum

Class that defines the permissible severities for a Vulnerability.

Note: See the CycloneDX schema: https://cyclonedx.org/docs/1.4/#type_severityType

---

NONE = 'none'

INFO = 'info'

LOW = 'low'

MEDIUM = 'medium'

HIGH = 'high'

CRITICAL = 'critical'

UNKNOWN = 'unknown'

static get_from_cvss_scores(*scores: Tuple[float, Ellipsis] | float | None*) → *VulnerabilitySeverity*

Derives the Severity of a Vulnerability from it's declared CVSS scores.

Args:
scores: A *tuple* of CVSS scores. CVSS scoring system allows for up to three separate scores.

Returns:
Always returns an instance of *VulnerabilitySeverity*.

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

rfind()

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

rindex()

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

rjust()

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.vulnerability.**VulnerabilityRating**(*, *source:* VulnerabilitySource | *None* = *None*, *score:* *decimal.Decimal* | *None* = *None*, *severity:* VulnerabilitySeverity | *None* = *None*, *method:* VulnerabilityScoreSource | *None* = *None*, *vector:* *str* | *None* = *None*, *justification:* *str* | *None* = *None*)

Class that models the *ratingType* complex element CycloneDX core schema.

This class previously modelled the *scoreType* complexe type in the schema extension used prior to schema version 1.4 - see https://github.com/CycloneDX/specification/blob/master/schema/ext/vulnerability-1.0.xsd.

Note: See *ratingType* in https://cyclonedx.org/docs/1.4/#ratingType

Warning: As part of implementing support for CycloneDX schema version 1.4, the three score types defined in the schema extension used prior to 1.4 have been deprecated. The deprecated *score_base* should loosely be equivalent to the new *score* in 1.4 schema. Both *score_impact* and *score_exploitability* are deprecated and removed as they are redundant if you have the vector (the vector allows you to calculate the scores).

property source: *VulnerabilitySource* | **None**

The source that published the vulnerability.

property score: **decimal.Decimal** | **None**

The numerical score of the rating.

property severity: *VulnerabilitySeverity* | **None**

The textual representation of the severity that corresponds to the numerical score of the rating.

property method: *VulnerabilityScoreSource* | **None**

The risk scoring methodology/standard used.

property vector: **str** | **None**

The textual representation of the metric values used to score the vulnerability - also known as the vector.

property justification: **str** | **None**

An optional reason for rating the vulnerability as it was.

class cyclonedx.model.vulnerability.**VulnerabilityCredits**(*, *organizations: Iterable[*cyclonedx.model.contact.OrganizationalEntity*]* | *None* = *None*, *individuals: Iterable[*cyclonedx.model.contact.OrganizationalContact*]* | *None* = *None*)

Class that models the *credits* of *vulnerabilityType* complex type in the CycloneDX schema (version >= 1.4).

This class also provides data support for schema versions < 1.4 where Vulnerabilites were possible through a schema extension (in XML only).

Note: See the CycloneDX schema: https://cyclonedx.org/docs/1.4/#type_vulnerabilityType

---

property organizations: **SortedSet[***OrganizationalEntity***]**

The organizations credited with vulnerability discovery.

Returns:
Set of *OrganizationalEntity*

property individuals: **SortedSet[***OrganizationalContact***]**

The individuals, not associated with organizations, that are credited with vulnerability discovery.

Returns:
Set of *OrganizationalContact*

class cyclonedx.model.vulnerability.**Vulnerability**(*, *bom_ref: str* | cyclonedx.model.bom_ref.BomRef | *None* = *None*, *id: str* | *None* = *None*, *source:* VulnerabilitySource | *None* = *None*, *references: Iterable[*VulnerabilityReference*]* | *None* = *None*, *ratings: Iterable[*VulnerabilityRating*]* | *None* = *None*, *cwes: Iterable[int]* | *None* = *None*, *description: str* | *None* = *None*, *detail: str* | *None* = *None*, *recommendation: str* | *None* = *None*, *advisories: Iterable[*VulnerabilityAdvisory*]* | *None* = *None*, *created: datetime.datetime* | *None* = *None*, *published: datetime.datetime* | *None* = *None*, *updated: datetime.datetime* | *None* = *None*, *credits:* VulnerabilityCredits | *None* = *None*, *tools: Iterable[*cyclonedx.model.Tool*]* | *None* = *None*, *analysis:* VulnerabilityAnalysis | *None* = *None*, *affects: Iterable[*BomTarget*]* | *None* = *None*, *properties: Iterable[*cyclonedx.model.Property*]* | *None* = *None*)

Class that models the *vulnerabilityType* complex type in the CycloneDX schema (version >= 1.4).

This class also provides data support for schema versions < 1.4 where Vulnerabilites were possible through a schema extension (in XML only).

Note: See the CycloneDX schema: https://cyclonedx.org/docs/1.4/#type_vulnerabilityType

property bom_ref: *cyclonedx.model.bom_ref.BomRef*

Get the unique reference for this Vulnerability in this BOM.

If a value was not provided in the constructor, a UUIDv4 will have been assigned.

Returns:
BomRef

property id: `str | None`

The identifier that uniquely identifies the vulnerability. For example: CVE-2021-39182.

Returns:
str if set else *None*

property source: *VulnerabilitySource* `| None`

The source that published the vulnerability.

Returns:
VulnerabilitySource if set else *None*

property references: `SortedSet[`*VulnerabilityReference*`]`

Zero or more pointers to vulnerabilities that are the equivalent of the vulnerability specified. Often times, the same vulnerability may exist in multiple sources of vulnerability intelligence, but have different identifiers. References provides a way to correlate vulnerabilities across multiple sources of vulnerability intelligence.

Returns:
Set of *VulnerabilityReference*

property ratings: `SortedSet[`*VulnerabilityRating*`]`

List of vulnerability ratings.

Returns:
Set of *VulnerabilityRating*

property cwes: `SortedSet[int]`

A list of CWE (Common Weakness Enumeration) identifiers.

Note: See https://cwe.mitre.org/

Returns:
Set of *int*

property description: `str | None`

A description of the vulnerability as provided by the source.

Returns:
str if set else *None*

property detail: str | None

If available, an in-depth description of the vulnerability as provided by the source organization. Details often include examples, proof-of-concepts, and other information useful in understanding root cause.

Returns:
str if set else *None*

property recommendation: str | None

Recommendations of how the vulnerability can be remediated or mitigated.

Returns:
str if set else *None*

property advisories: SortedSet[*VulnerabilityAdvisory*]

Advisories relating to the Vulnerability.

Returns:
Set of *VulnerabilityAdvisory*

property created: datetime.datetime | None

The date and time (timestamp) when the vulnerability record was created in the vulnerability database.

Returns:
datetime if set else *None*

property published: datetime.datetime | None

The date and time (timestamp) when the vulnerability record was first published.

Returns:
datetime if set else *None*

property updated: datetime.datetime | None

The date and time (timestamp) when the vulnerability record was last updated.

Returns:
datetime if set else *None*

property credits: *VulnerabilityCredits* | None

Individuals or organizations credited with the discovery of the vulnerability.

Returns:
VulnerabilityCredits if set else *None*

property tools: SortedSet[*Tool*]

The tool(s) used to identify, confirm, or score the vulnerability.

Returns:
Set of *Tool*

property analysis: *VulnerabilityAnalysis* | None

Analysis of the Vulnerability in your context.

Returns:
VulnerabilityAnalysis if set else *None*

property affects: SortedSet[*BomTarget*]

The components or services that are affected by the vulnerability.

Returns:
Set of *BomTarget*

property properties: SortedSet[*Property*]

Provides the ability to document properties in a key/value store. This provides flexibility to include data not officially supported in the standard without having to use additional namespaces or create extensions.

Return:
Set of *Property*

Package Contents

Classes

DataFlow	This is our internal representation of the dataFlowType simple type within the CycloneDX standard.
DataClassification	This is our internal representation of the *dataClassificationType* complex type within the CycloneDX standard.
Encoding	This is our internal representation of the encoding simple type within the CycloneDX standard.
AttachedText	This is our internal representation of the *attachedTextType* complex type within the CycloneDX standard.
HashAlgorithm	This is our internal representation of the hashAlg simple type within the CycloneDX standard.
HashType	This is our internal representation of the hashType complex type within the CycloneDX standard.
ExternalReferenceType	Enum object that defines the permissible 'types' for an External Reference according to the CycloneDX schema.
XsUri	Helper class that allows us to perform validation on data strings that are defined as xs:anyURI
ExternalReference	This is our internal representation of an ExternalReference complex type that can be used in multiple places within
Property	This is our internal representation of *propertyType* complex type that can be used in multiple places within
NoteText	This is our internal representation of the Note.text complex type that can be used in multiple places within
Note	This is our internal representation of the Note complex type that can be used in multiple places within
Tool	This is our internal representation of the *toolType* complex type within the CycloneDX standard.
IdentifiableAction	This is our internal representation of the *identifiableActionType* complex type.
Copyright	This is our internal representation of the *copyrightsType* complex type.

Attributes

ThisTool

class cyclonedx.model.**DataFlow**

Bases: str, enum.Enum

This is our internal representation of the dataFlowType simple type within the CycloneDX standard.

Note: See the CycloneDX Schema: https://cyclonedx.org/docs/1.4/xml/#type_dataFlowType

INBOUND = 'inbound'

OUTBOUND = 'outbound'

BI_DIRECTIONAL = 'bi-directional'

UNKNOWN = 'unknown'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

`isidentifier()`

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as “def” or “class”.

`islower()`

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

`isnumeric()`

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

`isprintable()`

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

`isspace()`

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

`istitle()`

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

`isupper()`

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

`join()`

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: ‘.’.join([‘ab’, ‘pq’, ‘rs’]) -> ‘ab.pq.rs’

`ljust()`

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

`lower()`

Return a copy of the string converted to lowercase.

`lstrip()`

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

rfind()

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

rindex()

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

rjust()

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

rpartition()

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

rsplit()

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

`translate()`

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.**DataClassification**(*, *flow:* DataFlow, *classification: str*)

This is our internal representation of the *dataClassificationType* complex type within the CycloneDX standard.

DataClassification might be deprecated since CycloneDX 1.5, but it is not deprecated in this library. In fact, this library will try to provide a compatibility layer if needed.

Note: See the CycloneDX Schema for dataClassificationType: https://cyclonedx.org/docs/1.4/xml/#type_dataClassificationType

property **flow**: *DataFlow*

Specifies the flow direction of the data.

Valid values are: inbound, outbound, bi-directional, and unknown.

Direction is relative to the service.

- Inbound flow states that data enters the service
- Outbound flow states that data leaves the service
- Bi-directional states that data flows both ways
- Unknown states that the direction is not known

Returns:
DataFlow

property **classification**: **str**

Data classification tags data according to its type, sensitivity, and value if altered, stolen, or destroyed.

Returns:
str

class cyclonedx.model.**Encoding**

Bases: str, enum.Enum

This is our internal representation of the encoding simple type within the CycloneDX standard.

Note: See the CycloneDX Schema: https://cyclonedx.org/docs/1.4/#type_encoding

`BASE_64 = 'base64'`

`capitalize()`

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

`casefold()`

Return a version of the string suitable for caseless comparisons.

`center()`

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

`count()`

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

`encode()`

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

`endswith()`

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

`expandtabs()`

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

`find()`

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`format()`

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

`format_map()`

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as “def” or “class”.

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

`istitle()`

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

`isupper()`

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

`join()`

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

`ljust()`

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

`lower()`

Return a copy of the string converted to lowercase.

`lstrip()`

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

`partition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

`removeprefix()`

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

`removesuffix()`

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

`replace()`

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

splitlines()

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

startswith()

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

strip()

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

swapcase()

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.**AttachedText**(*, *content: str*, *content_type: str = DEFAULT_CONTENT_TYPE*, *encoding:* Encoding | *None = None*)

This is our internal representation of the *attachedTextType* complex type within the CycloneDX standard.

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.3/#type_attachedTextType

property content_type: str

Specifies the content type of the text. Defaults to text/plain if not specified.

Returns:
str

property encoding: *Encoding* | None

Specifies the optional encoding the text is represented in.

Returns:
Encoding if set else *None*

property content: str

The attachment data.

Proactive controls such as input validation and sanitization should be employed to prevent misuse of attachment text.

Returns:
str

DEFAULT_CONTENT_TYPE = 'text/plain'

class cyclonedx.model.**HashAlgorithm**

Bases: `str`, `enum.Enum`

This is our internal representation of the hashAlg simple type within the CycloneDX standard.

Note: See the CycloneDX Schema: https://cyclonedx.org/docs/1.3/#type_hashAlg

BLAKE2B_256 = 'BLAKE2b-256'

BLAKE2B_384 = 'BLAKE2b-384'

BLAKE2B_512 = 'BLAKE2b-512'

BLAKE3 = 'BLAKE3'

MD5 = 'MD5'

SHA_1 = 'SHA-1'

SHA_256 = 'SHA-256'

SHA_384 = 'SHA-384'

SHA_512 = 'SHA-512'

SHA3_256 = 'SHA3-256'

SHA3_384 = 'SHA3-384'

SHA3_512 = 'SHA3-512'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

`casefold()`

Return a version of the string suitable for caseless comparisons.

`center()`

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

`count()`

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

`encode()`

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

`endswith()`

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

`expandtabs()`

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

`find()`

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`format()`

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

`format_map()`

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

`index()`

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as “def” or “class”.

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()
: Return True if the string is an uppercase string, False otherwise.

  A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()
: Concatenate any number of strings.

  The string whose method is called is inserted in between each given string. The result is returned as a new string.

  Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()
: Return a left-justified string of length width.

  Padding is done using the specified fill character (default is a space).

lower()
: Return a copy of the string converted to lowercase.

lstrip()
: Return a copy of the string with leading whitespace removed.

  If chars is given and not None, remove characters in chars instead.

partition()
: Partition the string into three parts using the given separator.

  This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

  If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()
: Return a str with the given prefix string removed if present.

  If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

removesuffix()
: Return a str with the given suffix string removed if present.

  If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

replace()
: Return a copy with all occurrences of substring old replaced by new.

  count
  : Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

  If the optional argument count is given, only the first count occurrences are replaced.

rfind()
: S.rfind(sub[, start[, end]]) -> int

  Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

  Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

startswith()

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

strip()

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

swapcase()

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill()

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

name()

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.**HashType**(*, *alg:* HashAlgorithm, *content: str*)

This is our internal representation of the hashType complex type within the CycloneDX standard.

Note: See the CycloneDX Schema for hashType: https://cyclonedx.org/docs/1.3/#type_hashType

property alg: *HashAlgorithm*

Specifies the algorithm used to create the hash.

Returns:
HashAlgorithm

property content: str

Hash value content.

Returns:
str

static from_hashlib_alg(*hashlib_alg: str*, *content: str*) → *HashType*

Attempts to convert a hashlib-algorithm to our internal model classes.

Args:

hashlib_alg:
Hash algorith - like it is used by *hashlib*. Example: *sha256*.

content:
Hash value.

Raises:
UnknownHashTypeException if the algorithm of hash cannot be determined.

Returns:
An instance of *HashType*.

static from_composite_str(*composite_hash: str*) → *HashType*

Attempts to convert a string which includes both the Hash Algorithm and Hash Value and represent using our internal model classes.

Args:

composite_hash:
Composite Hash string of the format *HASH_ALGORITHM*:*HASH_VALUE*. Example: *sha256:806143ae5bfb6a3c6e736a764057db0e6a0e05e338b5630894a5f779cabb4f9b*.

Raises:
UnknownHashTypeException if the type of hash cannot be determined.

Returns:
An instance of *HashType*.

class cyclonedx.model.**ExternalReferenceType**

Bases: str, enum.Enum

Enum object that defines the permissible 'types' for an External Reference according to the CycloneDX schema.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.3/#type_externalReferenceType

ADVERSARY_MODEL = 'adversary-model'

ADVISORIES = 'advisories'

ATTESTATION = 'attestation'

BOM = 'bom'

BUILD_META = 'build-meta'

BUILD_SYSTEM = 'build-system'

CERTIFICATION_REPORT = 'certification-report'

```
    CHAT = 'chat'

    CODIFIED_INFRASTRUCTURE = 'codified-infrastructure'

    COMPONENT_ANALYSIS_REPORT = 'component-analysis-report'

    CONFIGURATION = 'configuration'

    DIGITAL_SIGNATURE = 'digital-signature'

    DISTRIBUTION = 'distribution'

    DISTRIBUTION_INTAKE = 'distribution-intake'

    DOCUMENTATION = 'documentation'

    DYNAMIC_ANALYSIS_REPORT = 'dynamic-analysis-report'

    ELECTRONIC_SIGNATURE = 'electronic-signature'

    EVIDENCE = 'evidence'

    EXPLOITABILITY_STATEMENT = 'exploitability-statement'

    FORMULATION = 'formulation'

    ISSUE_TRACKER = 'issue-tracker'

    LICENSE = 'license'

    LOG = 'log'

    MAILING_LIST = 'mailing-list'

    MATURITY_REPORT = 'maturity-report'

    MODEL_CARD = 'model-card'

    PENTEST_REPORT = 'pentest-report'

    POAM = 'poam'

    QUALITY_METRICS = 'quality-metrics'

    RELEASE_NOTES = 'release-notes'

    RFC_9166 = 'rfc-9116'

    RISK_ASSESSMENT = 'risk-assessment'

    RUNTIME_ANALYSIS_REPORT = 'runtime-analysis-report'

    SECURITY_CONTACT = 'security-contact'

    STATIC_ANALYSIS_REPORT = 'static-analysis-report'

    SOCIAL = 'social'

    SOURCE_DISTRIBUTION = 'source-distribution'

    SCM = 'vcs'
```

SUPPORT = 'support'

THREAT_MODEL = 'threat-model'

VCS = 'vcs'

VULNERABILITY_ASSERTION = 'vulnerability-assertion'

WEBSITE = 'website'

OTHER = 'other'

capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

casefold()

Return a version of the string suitable for caseless comparisons.

center()

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

count()

S.count(sub[, start[, end]]) -> int

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding
The encoding in which to encode the string.

errors
The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors.

endswith()

S.endswith(suffix[, start[, end]]) -> bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

expandtabs()

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

find()

S.find(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

format()

S.format(*args, **kwargs) -> str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

format_map()

S.format_map(mapping) -> str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

index()

S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

removeprefix()

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

`removesuffix()`

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

`replace()`

Return a copy with all occurrences of substring old replaced by new.

count
Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

`rfind()`

S.rfind(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

`rindex()`

S.rindex(sub[, start[, end]]) -> int

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

`rjust()`

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

`rpartition()`

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

`rsplit()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

`rstrip()`

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`split()`

Return a list of the substrings in the string, using sep as the separator string.

sep
The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including n r t f and spaces) and will discard empty strings from the result.

maxsplit
Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

`splitlines()`

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

`startswith()`

S.startswith(prefix[, start[, end]]) -> bool

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

`strip()`

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

`swapcase()`

Convert uppercase characters to lowercase and lowercase characters to uppercase.

`title()`

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

`translate()`

Replace each character in the string using the given translation table.

table
Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

`upper()`

Return a copy of the string converted to uppercase.

`zfill()`

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

`name()`

The name of the Enum member.

value()

The value of the Enum member.

class cyclonedx.model.**XsUri**(*uri: str*)

Bases: serializable.helpers.BaseHelper

Helper class that allows us to perform validation on data strings that are defined as xs:anyURI in CycloneDX schema.

Developers can just use this via *str(XsUri('https://www.google.com'))*.

Note: See XSD definition for xsd:anyURI: http://www.datypic.com/sc/xsd/t-xsd_anyURI.html See JSON Schema definition for iri-reference: https://tools.ietf.org/html/rfc3987

---

property uri: str

classmethod serialize(*o: Any*) → str

classmethod deserialize(*o: Any*) → *XsUri*

class cyclonedx.model.**ExternalReference**(*, *type:* ExternalReferenceType, *url:* XsUri, *comment: str | None = None*, *hashes: Iterable[*HashType*] | None = None*)

This is our internal representation of an ExternalReference complex type that can be used in multiple places within a CycloneDX BOM document.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.3/#type_externalReference

---

property url: *XsUri*

The URL to the external reference.

Returns:
XsUri

property comment: str | None

An optional comment describing the external reference.

Returns:
str if set else *None*

property type: *ExternalReferenceType*

Specifies the type of external reference.

There are built-in types to describe common references. If a type does not exist for the reference being referred to, use the "other" type.

Returns:
ExternalReferenceType

property hashes: SortedSet[*HashType*]

The hashes of the external reference (if applicable).

Returns:
Set of *HashType*

class cyclonedx.model.**Property**(*, *name: str*, *value: str*)

This is our internal representation of *propertyType* complex type that can be used in multiple places within a CycloneDX BOM document.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_propertyType

Specifies an individual property with a name and value.

property name: str

The name of the property.

Duplicate names are allowed, each potentially having a different value.

Returns:
str

property value: str

Value of this Property.

Returns:
str

class cyclonedx.model.**NoteText**(*, *content: str*, *content_type: str | None = None*, *encoding:* Encoding *| None = None*)

This is our internal representation of the Note.text complex type that can be used in multiple places within a CycloneDX BOM document.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_releaseNotesType

property content: str

Get the text content of this Note.

Returns:
str note content

property content_type: str | None

Get the content-type of this Note.

Defaults to 'text/plain' if one was not explicitly specified.

Returns:
str content-type

property encoding: *Encoding* | None

Get the encoding method used for the note's content.

Returns:
Encoding if set else *None*

DEFAULT_CONTENT_TYPE: str = 'text/plain'

class cyclonedx.model.**Note**(*, *text:* NoteText, *locale: str | None = None*)

This is our internal representation of the Note complex type that can be used in multiple places within a CycloneDX BOM document.

Note: See the CycloneDX Schema definition: https://cyclonedx.org/docs/1.4/xml/#type_releaseNotesType

@todo: Replace `NoteText` with `AttachedText`?

property text: *NoteText*

Specifies the full content of the release note.

Returns:
NoteText

property locale: `str | None`

Get the ISO locale of this Note.

The ISO-639 (or higher) language code and optional ISO-3166 (or higher) country code.

Examples include: “en”, “en-US”, “fr” and “fr-CA”.

Returns:
str locale if set else *None*

class cyclonedx.model.**Tool**(*, *vendor: str | None = None*, *name: str | None = None*, *version: str | None = None*, *hashes: Iterable[HashType] | None = None*, *external_references: Iterable[ExternalReference] | None = None*)

This is our internal representation of the *toolType* complex type within the CycloneDX standard.

Tool(s) are the things used in the creation of the CycloneDX document.

Tool might be deprecated since CycloneDX 1.5, but it is not deprecated in this library. In fact, this library will try to provide a compatibility layer if needed.

Note: See the CycloneDX Schema for toolType: https://cyclonedx.org/docs/1.3/#type_toolType

property vendor: `str | None`

The name of the vendor who created the tool.

Returns:
str if set else *None*

property name: `str | None`

The name of the tool.

Returns:
str if set else *None*

property version: `str | None`

The version of the tool.

Returns:
str if set else *None*

property hashes: `SortedSet[HashType]`

The hashes of the tool (if applicable).

Returns:
Set of *HashType*

property external_references: `SortedSet[ExternalReference]`

External References provides a way to document systems, sites, and information that may be relevant but which are not included with the BOM.

Returns:
Set of *ExternalReference*

class cyclonedx.model.**IdentifiableAction**(*, *timestamp: datetime.datetime | None = None*, *name: str | None = None*, *email: str | None = None*)

This is our internal representation of the *identifiableActionType* complex type.

Note: See the CycloneDX specification: https://cyclonedx.org/docs/1.4/xml/#type_identifiableActionType

property timestamp: datetime.datetime | None

The timestamp in which the action occurred.

Returns:
datetime if set else *None*

property name: str | None

The name of the individual who performed the action.

Returns:
str if set else *None*

property email: str | None

The email address of the individual who performed the action.

Returns:
str if set else *None*

class cyclonedx.model.**Copyright**(*, *text: str*)

This is our internal representation of the *copyrightsType* complex type.

Note: See the CycloneDX specification: https://cyclonedx.org/docs/1.4/xml/#type_copyrightsType

property text: str

Copyright statement.

Returns:
str if set else *None*

cyclonedx.model.**ThisTool**

cyclonedx.output

Set of classes and methods for outputting our libraries internal Bom model to CycloneDX documents in varying formats and according to different versions of the CycloneDX schema standard.

Submodules

cyclonedx.output.json

Module Contents

Classes

Json	Helper class that provides a standard way to create an ABC using
JsonV1Dot0	Helper class that provides a standard way to create an ABC using
JsonV1Dot1	Helper class that provides a standard way to create an ABC using
JsonV1Dot2	Helper class that provides a standard way to create an ABC using
JsonV1Dot3	Helper class that provides a standard way to create an ABC using
JsonV1Dot4	Helper class that provides a standard way to create an ABC using
JsonV1Dot5	Helper class that provides a standard way to create an ABC using
JsonV1Dot6	Helper class that provides a standard way to create an ABC using

Attributes

BY_SCHEMA_VERSION

class cyclonedx.output.json.**Json**(*bom:* cyclonedx.model.bom.Bom)

Bases: *cyclonedx.output.BaseOutput*, *cyclonedx.schema.schema.BaseSchemaVersion*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: Literal[cyclonedx.schema.OutputFormat.JSON]

property generated: bool

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_bom() → *cyclonedx.model.bom.Bom*

set_bom(*bom:* cyclonedx.model.bom.Bom) → None

output_to_file(*filename: str*, *allow_overwrite: bool = False*, *, *indent: int | str | None = None*, ***kwargs: Any*) → None

class cyclonedx.output.json.**JsonV1Dot0**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Json*, *cyclonedx.schema.schema.SchemaVersion1Dot0*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: Literal[cyclonedx.schema.OutputFormat.JSON]

property generated: bool

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_bom() → *cyclonedx.model.bom.Bom*

set_bom(*bom:* cyclonedx.model.bom.Bom) → None

output_to_file(*filename: str*, *allow_overwrite: bool = False*, *, *indent: int | str | None = None*, ***kwargs: Any*) → None

class cyclonedx.output.json.**JsonV1Dot1**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Json*, *cyclonedx.schema.schema.SchemaVersion1Dot1*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: Literal[cyclonedx.schema.OutputFormat.JSON]

property generated: bool

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_bom() → *cyclonedx.model.bom.Bom*

set_bom(*bom:* cyclonedx.model.bom.Bom) → None

output_to_file(*filename: str*, *allow_overwrite: bool = False*, *, *indent: int | str | None = None*, ***kwargs: Any*) → None

class cyclonedx.output.json.**JsonV1Dot2**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Json*, *cyclonedx.schema.schema.SchemaVersion1Dot2*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: Literal[cyclonedx.schema.OutputFormat.JSON]

property generated: bool

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_bom() → *cyclonedx.model.bom.Bom*

set_bom(*bom:* cyclonedx.model.bom.Bom) → None

output_to_file(*filename: str*, *allow_overwrite: bool = False*, *, *indent: int | str | None = None*, ***kwargs: Any*) → None

class cyclonedx.output.json.**JsonV1Dot3**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Json*, *cyclonedx.schema.schema.SchemaVersion1Dot3*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: Literal[cyclonedx.schema.OutputFormat.JSON]

property generated: bool

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_bom() → *cyclonedx.model.bom.Bom*

set_bom(*bom:* cyclonedx.model.bom.Bom) → None

output_to_file(*filename: str*, *allow_overwrite: bool = False*, *, *indent: int | str | None = None*, ***kwargs: Any*) → None

class cyclonedx.output.json.**JsonV1Dot4**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Json*, *cyclonedx.schema.schema.SchemaVersion1Dot4*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: Literal[cyclonedx.schema.OutputFormat.JSON]

property generated: bool

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_bom() → *cyclonedx.model.bom.Bom*

set_bom(*bom:* cyclonedx.model.bom.Bom) → None

output_to_file(*filename: str*, *allow_overwrite: bool = False*, *, *indent: int | str | None = None*, ***kwargs: Any*) → None

class cyclonedx.output.json.**JsonV1Dot5**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Json*, *cyclonedx.schema.schema.SchemaVersion1Dot5*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: Literal[cyclonedx.schema.OutputFormat.JSON]

property generated: bool

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_bom() → *cyclonedx.model.bom.Bom*

set_bom(*bom:* cyclonedx.model.bom.Bom) → None

output_to_file(*filename: str*, *allow_overwrite: bool = False*, ***, *indent: int | str | None = None*, ***kwargs: Any*) → None

class cyclonedx.output.json.**JsonV1Dot6**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Json*, *cyclonedx.schema.schema.SchemaVersion1Dot6*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version: ***cyclonedx.schema.SchemaVersion***

property output_format: **Literal[cyclonedx.schema.OutputFormat.JSON]**

property generated: **bool**

generate(*force_regeneration: bool = False*) → None

output_as_string(***, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_bom() → *cyclonedx.model.bom.Bom*

set_bom(*bom:* cyclonedx.model.bom.Bom) → None

output_to_file(*filename: str*, *allow_overwrite: bool = False*, ***, *indent: int | str | None = None*, ***kwargs: Any*) → None

cyclonedx.output.json.**BY_SCHEMA_VERSION: Dict[*cyclonedx.schema.SchemaVersion*, Type[*Json*]]**

cyclonedx.output.xml

Module Contents

Classes

Xml
XmlV1Dot0
XmlV1Dot1
XmlV1Dot2
XmlV1Dot3
XmlV1Dot4
XmlV1Dot5
XmlV1Dot6

Attributes

BY_SCHEMA_VERSION

class cyclonedx.output.xml.**Xml**(*bom:* cyclonedx.model.bom.Bom)

Bases: *cyclonedx.schema.schema.BaseSchemaVersion*, *cyclonedx.output.BaseOutput*

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: Literal[cyclonedx.schema.OutputFormat.XML]

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_target_namespace() → str

class cyclonedx.output.xml.**XmlV1Dot0**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Xml*, *cyclonedx.schema.schema.SchemaVersion1Dot0*

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: Literal[cyclonedx.schema.OutputFormat.XML]

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_target_namespace() → str

class cyclonedx.output.xml.**XmlV1Dot1**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Xml*, *cyclonedx.schema.schema.SchemaVersion1Dot1*

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: Literal[cyclonedx.schema.OutputFormat.XML]

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_target_namespace() → str

class cyclonedx.output.xml.**XmlV1Dot2**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Xml*, *cyclonedx.schema.schema.SchemaVersion1Dot2*

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: Literal[cyclonedx.schema.OutputFormat.XML]

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_target_namespace() → str

class cyclonedx.output.xml.**XmlV1Dot3**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Xml*, *cyclonedx.schema.schema.SchemaVersion1Dot3*

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: **Literal[cyclonedx.schema.OutputFormat.XML]**

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_target_namespace() → str

class cyclonedx.output.xml.**XmlV1Dot4**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Xml*, *cyclonedx.schema.schema.SchemaVersion1Dot4*

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: **Literal[cyclonedx.schema.OutputFormat.XML]**

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_target_namespace() → str

class cyclonedx.output.xml.**XmlV1Dot5**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Xml*, *cyclonedx.schema.schema.SchemaVersion1Dot5*

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: **Literal[cyclonedx.schema.OutputFormat.XML]**

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_target_namespace() → str

class cyclonedx.output.xml.**XmlV1Dot6**(*bom:* cyclonedx.model.bom.Bom)

Bases: *Xml*, *cyclonedx.schema.schema.SchemaVersion1Dot6*

property schema_version: *cyclonedx.schema.SchemaVersion*

property output_format: **Literal[cyclonedx.schema.OutputFormat.XML]**

generate(*force_regeneration: bool = False*) → None

output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

get_target_namespace() → str

cyclonedx.output.xml.**BY_SCHEMA_VERSION: Dict[*cyclonedx.schema.SchemaVersion*, Type[*Xml*]]**

Package Contents

Classes

BaseOutput	Helper class that provides a standard way to create an ABC using
BomRefDiscriminator	

Functions

make_outputter(...)	Helper method to quickly get the correct output class/formatter.

class cyclonedx.output.**BaseOutput**(*bom:* cyclonedx.model.bom.Bom, ***kwargs: int*)

Bases: abc.ABC

Helper class that provides a standard way to create an ABC using inheritance.

abstract property schema_version: *cyclonedx.schema.SchemaVersion*

abstract property output_format: *cyclonedx.schema.OutputFormat*

property generated: bool

get_bom() → *cyclonedx.model.bom.Bom*

set_bom(*bom:* cyclonedx.model.bom.Bom) → None

abstract generate(*force_regeneration: bool = False*) → None

abstract output_as_string(*, *indent: int | str | None = None*, ***kwargs: Any*) → str

output_to_file(*filename: str*, *allow_overwrite: bool = False*, *, *indent: int | str | None = None*, ***kwargs: Any*) → None

cyclonedx.output.**make_outputter**(*bom:* cyclonedx.model.bom.Bom, *output_format: Literal[cyclonedx.schema.OutputFormat.JSON]*, *schema_version:* cyclonedx.schema.SchemaVersion) → *json.Json*

cyclonedx.output.**make_outputter**(*bom:* cyclonedx.model.bom.Bom, *output_format: Literal[cyclonedx.schema.OutputFormat.XML]*, *schema_version:* cyclonedx.schema.SchemaVersion) → *xml.Xml*

cyclonedx.output.**make_outputter**(*bom:* cyclonedx.model.bom.Bom, *output_format:* cyclonedx.schema.OutputFormat, *schema_version:* cyclonedx.schema.SchemaVersion) → *xml.Xml* | *json.Json*

Helper method to quickly get the correct output class/formatter.

Pass in your BOM and optionally an output format and schema version (defaults to XML and latest schema version).

Raises error when no instance could be made.

Parameters

- **bom** – Bom
- **output_format** – OutputFormat
- **schema_version** – SchemaVersion

Returns
BaseOutput

class cyclonedx.output.**BomRefDiscriminator**(*bomrefs: Iterable[cyclonedx.model.bom_ref.BomRef]*, *prefix: str = 'BomRef'*)

discriminate() → None

reset() → None

classmethod from_bom(*bom:* cyclonedx.model.bom.Bom, *prefix: str = 'BomRef'*) → *BomRefDiscriminator*

cyclonedx.schema

Submodules

cyclonedx.schema.schema

Module Contents

Classes

BaseSchemaVersion	Helper class that provides a standard way to create an ABC using
SchemaVersion1Dot6	Helper class that provides a standard way to create an ABC using
SchemaVersion1Dot5	Helper class that provides a standard way to create an ABC using
SchemaVersion1Dot4	Helper class that provides a standard way to create an ABC using
SchemaVersion1Dot3	Helper class that provides a standard way to create an ABC using
SchemaVersion1Dot2	Helper class that provides a standard way to create an ABC using
SchemaVersion1Dot1	Helper class that provides a standard way to create an ABC using
SchemaVersion1Dot0	Helper class that provides a standard way to create an ABC using

Attributes

SCHEMA_VERSIONS

class cyclonedx.schema.schema.**BaseSchemaVersion**

Bases: abc.ABC, serializable.ViewType

Helper class that provides a standard way to create an ABC using inheritance.

abstract property schema_version_enum: *cyclonedx.schema.SchemaVersion*

get_schema_version() → str

class cyclonedx.schema.schema.**SchemaVersion1Dot6**

Bases: *BaseSchemaVersion*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version_enum: Literal[cyclonedx.schema.SchemaVersion.V1_6]

get_schema_version() → str

class cyclonedx.schema.schema.**SchemaVersion1Dot5**

Bases: *BaseSchemaVersion*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version_enum: Literal[cyclonedx.schema.SchemaVersion.V1_5]

get_schema_version() → str

class cyclonedx.schema.schema.**SchemaVersion1Dot4**

Bases: *BaseSchemaVersion*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version_enum: Literal[cyclonedx.schema.SchemaVersion.V1_4]

get_schema_version() → str

class cyclonedx.schema.schema.**SchemaVersion1Dot3**

Bases: *BaseSchemaVersion*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version_enum: Literal[cyclonedx.schema.SchemaVersion.V1_3]

get_schema_version() → str

class cyclonedx.schema.schema.**SchemaVersion1Dot2**

Bases: *BaseSchemaVersion*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version_enum: Literal[cyclonedx.schema.SchemaVersion.V1_2]

get_schema_version() → str

class cyclonedx.schema.schema.**SchemaVersion1Dot1**

Bases: *BaseSchemaVersion*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version_enum: Literal[cyclonedx.schema.SchemaVersion.V1_1]

get_schema_version() → str

class cyclonedx.schema.schema.**SchemaVersion1Dot0**

Bases: *BaseSchemaVersion*

Helper class that provides a standard way to create an ABC using inheritance.

property schema_version_enum: Literal[cyclonedx.schema.SchemaVersion.V1_0]

get_schema_version() → str

cyclonedx.schema.schema.**SCHEMA_VERSIONS: Dict[*cyclonedx.schema.SchemaVersion*, Type[*BaseSchemaVersion*]]**

Package Contents

Classes

OutputFormat	Output formats.
SchemaVersion	Schema version.

class cyclonedx.schema.**OutputFormat**(**args*, ***kwds*)

Bases: enum.Enum

Output formats.

Cases are hashable.

Do not rely on the actual/literal values, just use enum cases, like so:
my_of = OutputFormat.XML

JSON

XML

name()
The name of the Enum member.

value()
The value of the Enum member.

class cyclonedx.schema.**SchemaVersion**(**args*, ***kwds*)

Bases: enum.Enum

Schema version.

Cases are hashable. Cases are comparable(!=,>=,>,==,<,<=)

Do not rely on the actual/literal values, just use enum cases, like so:
my_sv = SchemaVersion.V1_3

V1_6 = (1, 6)

V1_5 = (1, 5)

V1_4 = (1, 4)

V1_3 = (1, 3)

V1_2 = (1, 2)

V1_1 = (1, 1)

V1_0 = (1, 0)

classmethod from_version(*version: str*) → _SV

Return instance based of a version string - e.g. *1.4*

to_version() → str

Return as a version string - e.g. *1.4*

name()

The name of the Enum member.

value()

The value of the Enum member.

cyclonedx.serialization

Set of helper classes for use with `serializable` when conducting (de-)serialization.

Package Contents

Classes

BomRefHelper
PackageUrl
UrnUuidHelper
LicenseRepositoryHelper

class cyclonedx.serialization.**BomRefHelper**

Bases: `serializable.helpers.BaseHelper`

classmethod serialize(*o: Any*) → str | None

classmethod deserialize(*o: Any*) → *cyclonedx.model.bom_ref.BomRef*

class cyclonedx.serialization.**PackageUrl**

Bases: `serializable.helpers.BaseHelper`

classmethod serialize(*o: Any*) → str

classmethod deserialize(*o: Any*) → packageurl.PackageURL

class cyclonedx.serialization.**UrnUuidHelper**

Bases: serializable.helpers.BaseHelper

classmethod serialize(*o: Any*) → str

classmethod deserialize(*o: Any*) → uuid.UUID

class cyclonedx.serialization.**LicenseRepositoryHelper**

Bases: serializable.helpers.BaseHelper

classmethod json_normalize(*o:* cyclonedx.model.license.LicenseRepository, *, *view: Type[serializable.ViewType] | None*, ***__: Any*) → Any

classmethod json_denormalize(*o: List[Dict[str, Any]]*, ***__: Any*) → *cyclonedx.model.license.LicenseRepository*

classmethod xml_normalize(*o:* cyclonedx.model.license.LicenseRepository, *, *element_name: str*, *view: Type[serializable.ViewType] | None*, *xmlns: str | None*, ***__: Any*) → xml.etree.ElementTree.Element | None

classmethod xml_denormalize(*o: xml.etree.ElementTree.Element*, *default_ns: str | None*, ***__: Any*) → *cyclonedx.model.license.LicenseRepository*

cyclonedx.validation

Submodules

cyclonedx.validation.json

Module Contents

Classes

JsonValidator	Validator for CycloneDX documents in JSON format.
JsonStrictValidator	Strict validator for CycloneDX documents in JSON format.

class cyclonedx.validation.json.**JsonValidator**(*schema_version:* cyclonedx.schema.SchemaVersion)

Bases: _BaseJsonValidator, *cyclonedx.validation.BaseSchemabasedValidator*, *cyclonedx.validation.SchemabasedValidator*

Validator for CycloneDX documents in JSON format.

property output_format: Literal[cyclonedx.schema.OutputFormat.JSON]

get the format.

property schema_version: *cyclonedx.schema.SchemaVersion*

get the schema version.

validate_str(*data: str*) → *ValidationError* | None

Validate a string

Parameters
data – the data string to validate

Returns
validation error

Retval None
if *data* is valid

Retval ValidationError
if *data* is invalid

class cyclonedx.validation.json.**JsonStrictValidator**(*schema_version:* cyclonedx.schema.SchemaVersion)

Bases: _BaseJsonValidator, *cyclonedx.validation.BaseSchemabasedValidator*, *cyclonedx.validation.SchemabasedValidator*

Strict validator for CycloneDX documents in JSON format.

In contrast to *JsonValidator*, the document must not have additional or unknown JSON properties.

property output_format: Literal[cyclonedx.schema.OutputFormat.JSON]

get the format.

property schema_version: *cyclonedx.schema.SchemaVersion*

get the schema version.

validate_str(*data: str*) → *ValidationError* | None

Validate a string

Parameters
data – the data string to validate

Returns
validation error

Retval None
if *data* is valid

Retval ValidationError
if *data* is invalid

cyclonedx.validation.model

cyclonedx.validation.xml

Module Contents

Classes

XmlValidator	Validator for CycloneDX documents in XML format.

class cyclonedx.validation.xml.**XmlValidator**(*schema_version:* cyclonedx.schema.SchemaVersion)

Bases: _BaseXmlValidator, *cyclonedx.validation.BaseSchemabasedValidator*, *cyclonedx.validation.SchemabasedValidator*

Validator for CycloneDX documents in XML format.

property output_format: Literal[cyclonedx.schema.OutputFormat.XML]

Package Contents

Classes

ValidationError	Validation failed with this specific error.
SchemabasedValidator	Schema-based Validator protocol
BaseSchemabasedValidator	Base Schema-based Validator

Functions

make_schemabased_validator(...)	get the default Schema-based Validator for a certain :class:OutputFormat.

class cyclonedx.validation.**ValidationError**(*data: Any*)

Validation failed with this specific error.

Use *data* to access the content.

data: Any

class cyclonedx.validation.**SchemabasedValidator**

Bases: Protocol

Schema-based Validator protocol

validate_str(*data: str*) → *ValidationError* | None

Validate a string

Parameters
data – the data string to validate

Returns
validation error

Retval None
if *data* is valid

Retval ValidationError
if *data* is invalid

class cyclonedx.validation.**BaseSchemabasedValidator**(*schema_version:* cyclonedx.schema.SchemaVersion)

Bases: abc.ABC, *SchemabasedValidator*

Base Schema-based Validator

property schema_version: *cyclonedx.schema.SchemaVersion*

get the schema version.

abstract property output_format: *cyclonedx.schema.OutputFormat*

get the format.

validate_str(*data: str*) → *ValidationError* | None

Validate a string

Parameters
data – the data string to validate

Returns
validation error

Retval None
if *data* is valid

Retval ValidationError
if *data* is invalid

cyclonedx.validation.**make_schemabased_validator**(*output_format: Literal[cyclonedx.schema.OutputFormat.JSON]*, *schema_version:* cyclonedx.schema.SchemaVersion) → *json.JsonValidator*

cyclonedx.validation.**make_schemabased_validator**(*output_format: Literal[cyclonedx.schema.OutputFormat.XML]*, *schema_version:* cyclonedx.schema.SchemaVersion) → *xml.XmlValidator*

cyclonedx.validation.**make_schemabased_validator**(*output_format:* cyclonedx.schema.OutputFormat, *schema_version:* cyclonedx.schema.SchemaVersion) → *json.JsonValidator* | *xml.XmlValidator*

get the default Schema-based Validator for a certain :class:`OutputFormat`.

Raises error when no instance could be made.

7.1.2 Submodules

cyclonedx.spdx

Module Contents

Functions

is_supported_id(→ bool)	Validate a SPDX-ID according to current spec.
fixup_id(→ Optional[str])	Fixup a SPDX-ID.
is_compound_expression(→ bool)	Validate compound expression.

cyclonedx.spdx.**is_supported_id**(*value: str*) → bool

Validate a SPDX-ID according to current spec.

cyclonedx.spdx.**fixup_id**(*value: str*) → str | None

Fixup a SPDX-ID.

Returns
repaired value string, or *None* if fixup was unable to help.

cyclonedx.spdx.**is_compound_expression**(*value: str*) → bool

Validate compound expression.

Note: Utilizes license-expression library to validate SPDX compound expression according to SPDX license expression spec.

PYTHON MODULE INDEX

C

INDEX

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z