Estuary API Documentation

Matt Prahl, Yash Nanavati, Sarah Rieger

Jan 28, 2020

Contents

Build Status Docs Status

CHAPTER 1

Getting Started

1.1 Overview

Estuary visualizes the story an artifact takes in the Red Hat build to release pipeline, with a focus on the automation of container rebuilds due to CVEs. This repository contains the API and scrapers for the Estuary front end.

1.2 Development

To setup a development environment:

- Create and activate a Python virtual environment (Python 3 is preferred)
- Install the API and its dependencies with:

```
$ python setup.py develop
```

- (Optional): Install the scrapers' dependencies with:

```
$ pip install -r scraper-requirements.txt
```

To start the development web server, run:

```
$ scripts/run-flask.sh
```

1.3 Run the Unit Tests

Since the unit tests require a running Neo4j instance, the tests are run in Docker containers using Docker Compose. The commands required to run the unit tests are abstracted in `scripts/run-tests.sh`. This script will create the Docker image required to run the tests based on `docker/Dockerfile-tests`, create a container with Neo4j,

create another container to run the tests based on the built Docker image, run the tests, and then delete the two created containers.

To install Docker and Docker Compose on Fedora, run:

```
$ sudo dnf install docker docker-compose
```

To start Docker, run:

```
$ sudo systemctl start docker
```

To run the tests, run:

```
$ sudo scripts/run-tests.sh
```

To run just a single test, you can run:

```
sudo scripts/run-tests.sh pytest-3 -vvv tests/test_file::test_name
```

1.4 Code Styling

The codebase conforms to the style enforced by `flake8` with the following exceptions:

- The maximum line length allowed is 100 characters instead of 80 characters

In addition to `flake8`, docstrings are also enforced by the plugin `flake8-docstrings` with the following exemptions:

- D100: Missing docstring in public module
- D104: Missing docstring in public package

The format of the docstrings should be in the Sphynx style such as:

```
Get a resource from Neo4j.

:param str resource: a resource name that maps to a neomodel class
:param str uid: the value of the UniqueIdProperty to query with
:return: a Flask JSON response
:rtype: flask.Response
:raises NotFound: if the item is not found
:raises ValidationError: if an invalid resource was requested
```

1.5 Code Documentation

To document new files, please check here.

1.6 Authorization

If authentication is enabled, Estuary can authorize users based on their employee type and a user whitelist configured through the membership of an LDAP group.

1.6.1 Employee Type

You may set the list of valid employee types with the configuration item `EMPLOYEE_TYPES`. These employee types map to the `employeeType` LDAP attribute of the user that is added to the OpenID Connect token received by Estuary.

1.6.2 Configuring the Whitelist

To configure a whitelist of users, they must be part of an LDAP group configured with Estuary. The following configuration items are required:

- `LDAP_URI` - the URI to the LDAP server to connect to in the format of `ldaps://server.domain.local`.
- `LDAP_EXCEPTIONS_GROUP_DN` - the distinguished name to the LDAP group acting as the whitelist.

The following configuration items are optional:

- `LDAP_CA_CERTIFICATE` - the path to the CA certificate that signed the certificate used by the LDAP server. This only applies if you are using LDAPS. This defaults to `/etc/pki/tls/certs/ca-bundle.crt`.
- `LDAP_GROUP_MEMBERSHIP_ATTRIBUTE` - the LDAP attribute that represents a user in the group. This defaults to `uniqueMember`.

CHAPTER 2

Modules Documentation

2.1 API

estuary.api.v1.**about**()
Display general information about the app.

Return type flask.Response

estuary.api.v1.**get_artifact_relationships**(**args*, ***kwargs*)
Get one-to-many relationships of a particular artifact.

Parameters

- **resource** (*str*) – a resource name that maps to a neomodel class
- **uid** (*str*) – the value of the UniqueIdProperty to query with
- **relationship** (*str*) – relationship to expand

Returns a Flask JSON response

Return type flask.Response

Raises

- **NotFound** – if the item is not found
- **ValidationError** – if an invalid resource/relationship was requested

estuary.api.v1.**get_recent_stories**(**args*, ***kwargs*)
Get stories that were most recently updated, by their artifact type.

estuary.api.v1.**get_resource**(**args*, ***kwargs*)
Get a resource from Neo4j.

Parameters

- **resource** (*str*) – a resource name that maps to a neomodel class
- **uid** (*str*) – the value of the UniqueIdProperty to query with

Returns a Flask JSON response

Return type flask.Response

Raises

- **NotFound** – if the item is not found
- **ValidationError** – if an invalid resource was requested

estuary.api.v1.**get_resource_all_stories**(**args*, ***kwargs*)
Get all unique stories of an artifact from Neo4j.

Parameters

- **resource** (*str*) – a resource name that maps to a neomodel class
- **uid** (*str*) – the value of the UniqueIdProperty to query with

Returns a Flask JSON response

Return type flask.Response

Raises

- **NotFound** – if the item is not found
- **ValidationError** – if an invalid resource was requested

estuary.api.v1.**get_resource_story**(**args*, ***kwargs*)
Get the story of a resource from Neo4j.

Parameters

- **resource** (*str*) – a resource name that maps to a neomodel class
- **uid** (*str*) – the value of the UniqueIdProperty to query with

Returns a Flask JSON response

Return type flask.Response

Raises

- **NotFound** – if the item is not found
- **ValidationError** – if an invalid resource was requested

estuary.api.v1.**get_siblings**(**args*, ***kwargs*)
Get siblings of next/previous node that are correlated to the node in question.

Parameters

- **resource** (*str*) – a resource name that maps to a neomodel class
- **uid** (*str*) – the value of the UniqueIdProperty to query with

Returns a Flask JSON response

Return type flask.Response

Raises

- **NotFound** – if the item is not found
- **ValidationError** – if an invalid resource was requested

2.2 Models

2.2.1 Base

class estuary.models.base.**EstuaryStructuredNode**(**args*, ***kwargs*)

Base class for Estuary Neo4j models.

DoesNotExist

alias of neomodel.core.EstuaryStructuredNodeDoesNotExist

add_label(*new_label*)

Add a Neo4j label to an existing node.

Parameters **new_label** (*str*) – the new label to add to the node

static conditional_connect(*relationship*, *new_node*)

Wrap the connect and replace methods for conditional relationship handling.

Parameters

- **relationship** (*neomodel.RelationshipManager*) – a relationship to connect on
- **new_node** (*neomodel.StructuredNode*) – the node to create the relationship with

Raises **NotImplementedError** – if this method is called with a relationship of cardinality of one

display_name

Get intuitive (human readable) display name for the node.

classmethod find_or_none(*identifier*)

Find the node using the supplied identifier.

This method should be overridden if the node class accepts multiple types of identifiers. :param str identifier: the identifier to search the node by :return: the node or None :rtype: EstuaryStructuredNode or None

static inflate_results(*results*)

Inflate the results.

Parameters **results** (*str*) – results obtained from Neo4j

Returns a list of dictionaries containing serialized results received from Neo4j

Return type list

remove_label(*label*)

Remove a Neo4j label from an existing node.

Parameters **label** (*str*) – the label to be removed from the node

serialized

Convert a model to serialized form.

Returns a serialized form of the node

Return type dictionary

serialized_all

Generate a serialized form of the node that includes all its relationships.

Returns a serialized form of the node with relationships

Return type dictionary

Raises **RuntimeError** – if the label of a Neo4j node can't be mapped back to a neomodel class

timeline_datetime
Get the DateTime property used for the Estuary timeline.

timeline_timestamp
Get the DateTime property used for the Estuary timeline as a string.

unique_id_property
Get the name of the UniqueIdProperty for the node.

Returns a string containing name of the unique ID property of a node

Return type str

2.2.2 Bugzilla

class estuary.models.bugzilla.**BugzillaBug**(**args*, ***kwargs*)
Definition of a Bugzilla bug in Neo4j.

DoesNotExist
alias of neomodel.core.BugzillaBugDoesNotExist

assignee = <neomodel.relationship_manager.RelationshipDefinition object>

attached_advisories = <neomodel.relationship_manager.RelationshipDefinition object>

classification = <neomodel.properties.StringProperty object>

creation_time = <neomodel.properties.DateTimeProperty object>

display_name
Get intuitive (human readable) display name for the node.

classmethod find_or_none(*identifier*)
Find the node using the supplied identifier.

Parameters **identifier** (*str*) – the identifier to search the node by

Returns the node or None

Return type *EstuaryStructuredNode* or None

id_ = <neomodel.properties.UniqueIdProperty object>

modified_time = <neomodel.properties.DateTimeProperty object>

priority = <neomodel.properties.StringProperty object>

product_name = <neomodel.properties.StringProperty object>

product_version = <neomodel.properties.StringProperty object>

qa_contact = <neomodel.relationship_manager.RelationshipDefinition object>

related_by_commits = <neomodel.relationship_manager.RelationshipDefinition object>

reporter = <neomodel.relationship_manager.RelationshipDefinition object>

resolution = <neomodel.properties.StringProperty object>

resolved_by_commits = <neomodel.relationship_manager.RelationshipDefinition object>

reverted_by_commits = <neomodel.relationship_manager.RelationshipDefinition object>

severity = <neomodel.properties.StringProperty object>

short_description = <neomodel.properties.StringProperty object>

status = <neomodel.properties.StringProperty object>

target_milestone = <neomodel.properties.StringProperty object>

timeline_datetime
Get the DateTime property used for the Estuary timeline.

votes = <neomodel.properties.IntegerProperty object>

2.2.3 DistGit

class estuary.models.distgit.**DistGitBranch**(**args*, ***kwargs*)
Definition of a dist-git branch in Neo4j.

DoesNotExist
alias of neomodel.core.DistGitBranchDoesNotExist

commits = <neomodel.relationship_manager.RelationshipDefinition object>

contributors = <neomodel.relationship_manager.RelationshipDefinition object>

display_name
Get intuitive (human readable) display name for the node.

name = <neomodel.properties.StringProperty object>

repo_name = <neomodel.properties.StringProperty object>

repo_namespace = <neomodel.properties.StringProperty object>

repos = <neomodel.relationship_manager.RelationshipDefinition object>

class estuary.models.distgit.**DistGitCommit**(**args*, ***kwargs*)
Definition of a dist-git commit in Neo4j.

DoesNotExist
alias of neomodel.core.DistGitCommitDoesNotExist

author = <neomodel.relationship_manager.RelationshipDefinition object>

author_date = <neomodel.properties.DateTimeProperty object>

branches = <neomodel.relationship_manager.RelationshipDefinition object>

children = <neomodel.relationship_manager.RelationshipDefinition object>

commit_date = <neomodel.properties.DateTimeProperty object>

display_name
Get intuitive (human readable) display name for the node.

hash_ = <neomodel.properties.UniqueIdProperty object>

koji_builds = <neomodel.relationship_manager.RelationshipDefinition object>

log_message = <neomodel.properties.StringProperty object>

parent = <neomodel.relationship_manager.RelationshipDefinition object>

related_bugs = <neomodel.relationship_manager.RelationshipDefinition object>

repos = <neomodel.relationship_manager.RelationshipDefinition object>

resolved_bugs = <neomodel.relationship_manager.RelationshipDefinition object>

reverted_bugs = <neomodel.relationship_manager.RelationshipDefinition object>

timeline_datetime
Get the DateTime property used for the Estuary timeline.

class estuary.models.distgit.**DistGitRepo**(**args*, ***kwargs*)
Definition of a dist-git repo in Neo4j.

DoesNotExist
alias of neomodel.core.DistGitRepoDoesNotExist

branches = <neomodel.relationship_manager.RelationshipDefinition object>

commits = <neomodel.relationship_manager.RelationshipDefinition object>

contributors = <neomodel.relationship_manager.RelationshipDefinition object>

display_name
Get intuitive (human readable) display name for the node.

name = <neomodel.properties.StringProperty object>

namespace = <neomodel.properties.StringProperty object>

2.2.4 Errata

class estuary.models.errata.**Advisory**(**args*, ***kwargs*)
Definition of an Errata advisory in Neo4j.

class BuildAttachedRel(**args*, ***kwargs*)
Definition of a relationship between an Advisory and a KojiBuild attached to it.

time_attached = <neomodel.properties.DateTimeProperty object>

DoesNotExist
alias of neomodel.core.AdvisoryDoesNotExist

actual_ship_date = <neomodel.properties.DateTimeProperty object>

advisory_name = <neomodel.properties.StringProperty object>

assigned_to = <neomodel.relationship_manager.RelationshipDefinition object>

attached_bugs = <neomodel.relationship_manager.RelationshipDefinition object>

classmethod attached_build_time(*advisory*, *build*)
Get the time that a build related to the advisory was attached.

Parameters **build** (*node*) – a Neo4j node representing an attached build

Returns the time the build was attached

Return type datetime object

attached_builds = <neomodel.relationship_manager.RelationshipDefinition object>

content_types = <neomodel.properties.ArrayProperty object>

created_at = <neomodel.properties.DateTimeProperty object>

display_name
Get intuitive (human readable) display name for the node.

classmethod find_or_none(*identifier*)
Find the node using the supplied identifier.

Parameters **identifier** (*str*) – the identifier to search the node by

Returns the node or None

Return type *EstuaryStructuredNode* or None

id_ = <neomodel.properties.UniqueIdProperty object>

issue_date = <neomodel.properties.DateTimeProperty object>

product_name = <neomodel.properties.StringProperty object>

product_short_name = <neomodel.properties.StringProperty object>

release_date = <neomodel.properties.DateTimeProperty object>

reporter = <neomodel.relationship_manager.RelationshipDefinition object>

security_impact = <neomodel.properties.StringProperty object>

security_sla = <neomodel.properties.DateTimeProperty object>

state = <neomodel.properties.StringProperty object>

status_time = <neomodel.properties.DateTimeProperty object>

synopsis = <neomodel.properties.StringProperty object>

timeline_datetime
Get the DateTime property used for the Estuary timeline.

triggered_freshmaker_event = <neomodel.relationship_manager.RelationshipDefinition obj

update_date = <neomodel.properties.DateTimeProperty object>

class estuary.models.errata.**ContainerAdvisory**(**args*, ***kwargs*)
Definition of an Errata advisory with container builds attached in Neo4j.

DoesNotExist
alias of neomodel.core.ContainerAdvisoryDoesNotExist

2.2.5 Freshmaker

class estuary.models.freshmaker.**FreshmakerBuild**(**args*, ***kwargs*)
Definition of a Freshmaker build in Neo4j.

DoesNotExist
alias of neomodel.core.FreshmakerBuildDoesNotExist

build_id = <neomodel.properties.IntegerProperty object>

dep_on = <neomodel.properties.StringProperty object>

display_name
Get intuitive (human readable) display name for the node.

event = <neomodel.relationship_manager.RelationshipDefinition object>

id_ = <neomodel.properties.UniqueIdProperty object>

koji_builds = <neomodel.relationship_manager.RelationshipDefinition object>

name = <neomodel.properties.StringProperty object>

original_nvr = <neomodel.properties.StringProperty object>

rebuilt_nvr = <neomodel.properties.StringProperty object>

state = <neomodel.properties.IntegerProperty object>

state_name = <neomodel.properties.StringProperty object>

state_reason = <neomodel.properties.StringProperty object>

time_completed = <neomodel.properties.DateTimeProperty object>

time_submitted = <neomodel.properties.DateTimeProperty object>

type_ = <neomodel.properties.IntegerProperty object>

type_name = <neomodel.properties.StringProperty object>

url = <neomodel.properties.StringProperty object>

class estuary.models.freshmaker.**FreshmakerEvent**(**args*, ***kwargs*)

Definition of a Freshmaker event in Neo4j.

DoesNotExist

alias of neomodel.core.FreshmakerEventDoesNotExist

display_name

Get intuitive (human readable) display name for the node.

event_type_id = <neomodel.properties.IntegerProperty object>

id_ = <neomodel.properties.UniqueIdProperty object>

message_id = <neomodel.properties.StringProperty object>

requested_builds = <neomodel.relationship_manager.RelationshipDefinition object>

state = <neomodel.properties.IntegerProperty object>

state_name = <neomodel.properties.StringProperty object>

state_reason = <neomodel.properties.StringProperty object>

successful_koji_builds = <neomodel.relationship_manager.RelationshipDefinition object>

time_created = <neomodel.properties.DateTimeProperty object>

time_done = <neomodel.properties.DateTimeProperty object>

timeline_datetime

Get the DateTime property used for the Estuary timeline.

triggered_by_advisory = <neomodel.relationship_manager.RelationshipDefinition object>

2.2.6 Koji

class estuary.models.koji.**ContainerKojiBuild**(**args*, ***kwargs*)

A Neo4j definition of a build that represents a container build in Koji.

DoesNotExist

alias of neomodel.core.ContainerKojiBuildDoesNotExist

operator = <neomodel.properties.BooleanProperty object>

original_nvr = <neomodel.properties.StringProperty object>

triggered_by_freshmaker_event = <neomodel.relationship_manager.RelationshipDefinition

class estuary.models.koji.**KojiBuild**(**args*, ***kwargs*)
Definition of a Koji build in Neo4j.

DoesNotExist
alias of neomodel.core.KojiBuildDoesNotExist

advisories = <neomodel.relationship_manager.RelationshipDefinition object>

commit = <neomodel.relationship_manager.RelationshipDefinition object>

completion_time = <neomodel.properties.DateTimeProperty object>

creation_time = <neomodel.properties.DateTimeProperty object>

display_name
Get intuitive (human readable) display name for the node.

epoch = <neomodel.properties.StringProperty object>

extra = <neomodel.properties.StringProperty object>

classmethod find_or_none(*identifier*)
Find the node using the supplied identifier.

Parameters **identifier** (*str*) – the identifier to search the node by

Returns the node or None

Return type *EstuaryStructuredNode* or None

id_ = <neomodel.properties.UniqueIdProperty object>

module_builds = <neomodel.relationship_manager.RelationshipDefinition object>

name = <neomodel.properties.StringProperty object>

owner = <neomodel.relationship_manager.RelationshipDefinition object>

release = <neomodel.properties.StringProperty object>

start_time = <neomodel.properties.DateTimeProperty object>

state = <neomodel.properties.IntegerProperty object>

tags = <neomodel.relationship_manager.RelationshipDefinition object>

timeline_datetime
Get the DateTime property used for the Estuary timeline.

version = <neomodel.properties.StringProperty object>

class estuary.models.koji.**KojiTag**(**args*, ***kwargs*)
Definition of a Koji tag in Neo4j.

DoesNotExist
alias of neomodel.core.KojiTagDoesNotExist

builds = <neomodel.relationship_manager.RelationshipDefinition object>

display_name
Get intuitive (human readable) display name for the node.

id_ = <neomodel.properties.UniqueIdProperty object>

module_builds = <neomodel.relationship_manager.RelationshipDefinition object>

name = <neomodel.properties.StringProperty object>

class estuary.models.koji.**ModuleKojiBuild**(**args*, ***kwargs*)
A Neo4j definition of a build that represents a module build in Koji.

DoesNotExist
alias of neomodel.core.ModuleKojiBuildDoesNotExist

components = <neomodel.relationship_manager.RelationshipDefinition object>

content_koji_tag = <neomodel.relationship_manager.RelationshipDefinition object>

context = <neomodel.properties.StringProperty object>

mbs_id = <neomodel.properties.IntegerProperty object>

module_name = <neomodel.properties.StringProperty object>

module_stream = <neomodel.properties.StringProperty object>

module_version = <neomodel.properties.StringProperty object>

2.2.7 User

class estuary.models.user.**User**(**args*, ***kwargs*)
Definition of a generic user in Neo4j.

DoesNotExist
alias of neomodel.core.UserDoesNotExist

advisories_assigned = <neomodel.relationship_manager.RelationshipDefinition object>

advisories_reported = <neomodel.relationship_manager.RelationshipDefinition object>

bugs_assigned = <neomodel.relationship_manager.RelationshipDefinition object>

bugs_qa_contact_for = <neomodel.relationship_manager.RelationshipDefinition object>

bugs_reported = <neomodel.relationship_manager.RelationshipDefinition object>

display_name
Get intuitive (human readable) display name for the node.

distgit_authored_commits = <neomodel.relationship_manager.RelationshipDefinition objec

distgit_branches = <neomodel.relationship_manager.RelationshipDefinition object>

distgit_repos = <neomodel.relationship_manager.RelationshipDefinition object>

email = <neomodel.properties.StringProperty object>

koji_builds = <neomodel.relationship_manager.RelationshipDefinition object>

name = <neomodel.properties.StringProperty object>

username = <neomodel.properties.UniqueIdProperty object>

2.3 Scrapers

2.3.1 Base

class scrapers.base.**BaseScraper**(*teiid_user=None*, *teiid_password=None*, *kerberos=False*, *neo4j_user=u'neo4j'*, *neo4j_password=u'neo4j'*, *neo4j_server=u'localhost'*)

Base scraper class to standardize the main scraper functionality.

default_since = '2019-01-28'

default_until = '2020-01-29'

is_container_build(*build_info*)

Check whether a Koji build is a container build.

Parameters **build_info** (KojiBuild) – build info from Teiid

Returns boolean value indicating whether the build is a container build

Return type bool

is_module_build(*build_info*)

Check whether a Koji build is a module build.

Parameters **build_info** (KojiBuild) – build info from Teiid

Returns boolean value indicating whether the build is a module build

Return type bool

run(*since=None*)

Run the scraper.

Parameters **since** (*str*) – a datetime to start scraping data from

Raises **NotImplementedError** – if the function is not overridden

teiid_host = u'virtualdb.engineering.redhat.com'

teiid_port = 5432

2.3.2 Bugzilla

class scrapers.bugzilla.**BugzillaScraper**(*teiid_user=None*, *teiid_password=None*, *kerberos=False*, *neo4j_user=u'neo4j'*, *neo4j_password=u'neo4j'*, *neo4j_server=u'localhost'*)

Scrapes the Bugzilla tables in Teiid.

create_user_node(*email*)

Create a User node in Neo4j.

Parameters **email** (*str*) – the user's email

Returns User object

get_bugzilla_bugs(*start_date*, *end_date*)

Get the Buzilla bugs information from Teiid.

Parameters

- **start_date** (*datetime.datetime*) – when to start scraping data from

- **end_date** (*datetime.datetime*) – determines until when to scrape data

Returns list of dictionaries containing bug info

Return type list

run(*since=None*, *until=None*)
Run the Bugzilla scraper.

Parameters

- **since** (*str*) – a datetime to start scraping data from
- **until** (*str*) – a datetime to scrape data until

update_neo4j(*bugs*)
Update Neo4j with Bugzilla bugs information from Teiid.

Parameters **bugs** (*list*) – a list of dictionaries

2.3.3 DistGit

class scrapers.distgit.**DistGitScraper**(*teiid_user=None*, *teiid_password=None*, *kerberos=False*, *neo4j_user=u'neo4j'*, *neo4j_password=u'neo4j'*, *neo4j_server=u'localhost'*)
Scrapes the GitBZ tables in Teiid.

get_distgit_data(*since*, *until*)
Query Teiid for the dist-git commit and Bugzilla information.

Parameters

- **since** (*datetime.datetime*) – determines when to start the query
- **until** (*datetime.datetime*) – determines until when to scrape data

Returns a list of dictionaries

Return type list

run(*since=None*, *until=None*)
Run the dist-git scraper.

Parameters

- **since** (*str*) – a datetime to start scraping data from
- **until** (*str*) – a datetime to scrape data until

2.3.4 Errata

class scrapers.errata.**ErrataScraper**(*teiid_user=None*, *teiid_password=None*, *kerberos=False*, *neo4j_user=u'neo4j'*, *neo4j_password=u'neo4j'*, *neo4j_server=u'localhost'*)
Scrapes the Errata Tool tables in Teiid.

get_advisories(*since*, *until*)
Query Teiid for the Errata Tool advisories.

Parameters

- **since** (*datetime.datetime*) – determines when to start querying

- **until** (*datetime.datetime*) – determines until when to scrape data

Returns a list of dictionaries

Return type list

get_associated_builds(*advisory_id*)
Query Teiid to find the Brew builds associated with a specific advisory.

Parameters **advisory_id** (*int*) – the advisory ID

Returns a list of a dictionaries

Return type list

get_attached_bugs(*advisory_id*)
Query Teiid to find the Bugzilla bugs attached to a specific advisory.

Parameters **advisory_id** (*int*) – the advisory ID

Returns a list of a dictionaries

Return type list

get_koji_build(*build_id*)
Query Teiid to find the Koji build attached to a specific advisory.

Parameters **build_id** (*int*) – the build ID

Returns a list of a dictionaries

Return type list

run(*since=None*, *until=None*)
Run the Errata Tool scraper.

Parameters

- **since** (*str*) – a datetime to start scraping data from
- **until** (*str*) – a datetime to scrape data until

update_neo4j(*advisories*)
Update Neo4j with Errata Tool advisories from Teiid.

Parameters **advisories** (*list*) – a list of dictionaries of advisories

2.3.5 Freshmaker

class scrapers.freshmaker.**FreshmakerScraper**(*teiid_user=None*, *teiid_password=None*, *kerberos=False*, *neo4j_user=u'neo4j'*, *neo4j_password=u'neo4j'*, *neo4j_server=u'localhost'*)
Scrapes the Freshmaker API.

freshmaker_url = u'https://freshmaker.engineering.redhat.com/api/2/events/?per_page=50

get_koji_task_result(*task_id*)
Query Teiid for a Koji task's result attribute.

Parameters **task_id** (*int*) – the Koji task ID to query

Returns an XML string

Return type str

query_api_and_update_neo4j()
Scrape the Freshmaker API and upload the data to Neo4j.

Parameters **start_date** (*str*) – a datetime to start scraping data from

run(*since=None*, *until=None*)
Run the Freshmaker scraper.

Parameters

- **since** (*str*) – a datetime to start scraping data from
- **until** (*str*) – a datetime to scrape data until

2.3.6 Koji

class scrapers.koji.**KojiScraper**(*teiid_user=None*, *teiid_password=None*, *kerberos=False*, *neo4j_user=u'neo4j'*, *neo4j_password=u'neo4j'*, *neo4j_server=u'localhost'*)
Scrapes the Koji tables in Teiid.

get_build_info(*build_ids*)
Query Teiid for build info.

Parameters **build_ids** (*list*) – ID's of Koji builds

Returns a list of dictionaries

Return type list

get_build_tags(*build_id*)
Query Teiid for all tags a build is tagged in.

Parameters **build_id** (*int*) – the Koji build's ID

Returns a list of dictionaries

Return type list

get_koji_builds(*start_date*, *end_date*)
Query Teiid for Koji builds.

Parameters

- **start_date** (*datetime.datetime*) – determines when to start the query
- **end_date** (*datetime.datetime*) – determines until when to scrape data

Returns a list of dictionaries

Return type list

get_tag_info(*tag_name*)
Query Teiid for tag_id of a tag and build_ids associated to it.

Parameters **tag_name** (*str*) – tag name

Returns a list of dictionaries

Return type list

get_task(*task_id*)
Query Teiid for a Koji task.

Parameters **task_id** (*int*) – the Koji task ID to query

Returns a list of dictionaries

Return type list

run(*since=None*, *until=None*)
Run the Koji scraper.

Parameters

- **since** (*str*) – a datetime to start scraping data from
- **until** (*str*) – a datetime to scrape data until

update_neo4j(*builds*)
Update Neo4j with Koji build information from Teiid.

Parameters **builds** (*list*) – a list of dictionaries

2.3.7 Teiid

class scrapers.teiid.**Teiid**(*host*, *port*, *username*, *password*)
Abstracts interfacing with Teiid to simplify connections and queries.

get_connection(*db_name*, *force_new=False*, *retry=None*)
Return an existing psycopg2 connection and establish it if needed.

Parameters

- **db_name** (*str*) – the database name to get a connection to
- **force_new** (*bool*) – forces a new database connection even if one already exists
- **retry** (*int*) – the number of times to retry a failed connection. If this is not set, then the Teiid connection attempt will be repeated until it is successful.

Returns a connection to Teiid

Return type psycopg2 connection

query(*sql*, *db=u'public'*, *retry=None*)
Send the SQL query to Teiid and return the rows as a list.

Parameters

- **sql** (*str*) – the SQL query to send to the database
- **db** (*str*) – the database name to query on
- **retry** (*int*) – the number of times to retry a failed query. If this is not set, then the Teiid query will be repeated until it is successful.

Returns a list of rows from Teiid. Each row is a dictionary with the column headers as the keys.

Return type list

2.3.8 Utils

scrapers.utils.**retry_session**()
Create a python-requests session that retries on connection failures.

Returns a configured session object

Return type requests.Session

2.4 Utils

2.4.1 General

estuary.utils.general.**get_neo4j_node**(*resource_name*, *uid*)
Get a Neo4j node based on a label and unique identifier.

Parameters

- **resource_name** (*str*) – a neomodel model label
- **uid** (*str*) – a string of the unique identifier defined in the neomodel model

Returns a neomodel model object

Raises **ValidationError** – if the requested resource doesn't exist or doesn't have a UniqueIdProperty

estuary.utils.general.**inflate_node**(*result*)
Inflate a Neo4j result to a neomodel model object.

Parameters **result** (*neo4j.v1.types.Node*) – a node from a cypher query result

Returns a model (EstuaryStructuredNode) object

estuary.utils.general.**login_required**(*f*)
Decorate a Flask route to validate a token if authentication is enabled.

Parameters **f** (*function*) – the function to wrap

Returns the wrapper function

Return type function

estuary.utils.general.**str_to_bool**(*item*)
Convert a string to a boolean.

Parameters **item** (*str*) – string to parse

Returns a boolean equivalent

Return type boolean

estuary.utils.general.**timestamp_to_date**(*timestamp*)
Convert a string timestamp to a date object.

Parameters **timestamp** (*str*) – a generic or ISO-8601 timestamp

Returns date object of the timestamp

Return type datetime.date

Raises **ValueError** – if the timestamp is an unsupported or invalid format

estuary.utils.general.**timestamp_to_datetime**(*timestamp*)
Convert a string timestamp to a datetime object.

Parameters **timestamp** (*str*) – a generic or ISO-8601 timestamp

Returns datetime object of the timestamp

Return type datetime.datetime

Raises **ValueError** – if the timestamp is an unsupported or invalid format

2.4.2 Story

class estuary.utils.story.**BaseStoryManager**
A class containing utility methods to create a story for an artifact.

format_story_results(*results*, *requested_item*)
Format story results from Neo4j to the API format.

Parameters

- **results** (*list*) – nodes in a story/path
- **requested_item** (EstuaryStructuredNode) – item requested by the user

Returns results in API format

Return type dict

get_sibling_nodes(*siblings_node_label*, *story_node*, *count=False*)
Return sibling nodes with the label siblings_node_label that are related to story_node.

Parameters

- **siblings_node_label** (*str*) – node label for which the siblings count is to be calculated
- **story_node** (EstuaryStructuredNode) – node in the story that has the desired relationships with the siblings (specified with siblings_node_label)
- **count** (*bool*) – determines if only count of sibling nodes should be returned or the nodes themselves

Returns siblings count of curr_node | sibling nodes

Return type int | EstuaryStructuredNode

get_sibling_nodes_count(*results*, *reverse=False*)
Iterate through the results and yield correlated nodes.

Parameters

- **results** (*list*) – contains inflated results from Neo4j
- **reverse** (*bool*) – determines the direction the story is traversed in (i.e. forward/backward)

Returns yield the results count (int) received from Neo4j

Return type generator

static get_siblings_description(*story_node_display_name*, *story_node_story_flow*, *backward*)
Generate a description of the siblings.

Parameters

- **story_node_display_name** (*string*) – the preformatted name to be displayed for the story node
- **story_node_story_flow** (*dict*) – has forward and backward relationships of the story node
- **backward** (*bool*) – determines the relationship direction the story node has with the siblings in the story

Returns returns the appropriate siblings title

Return type string

static get_story_manager(*item*, *config*, *limit=False*)
Select which story flow to follow.

Parameters

- **item** (*node*) – a Neo4j node whose story is requested by the user
- **config** (*flask.config.Config*) – flask config
- **limit** (*bool*) – specifies if LIMIT keyword should be added to the created cypher query

Returns instance of one of the story manager classes

Return type ModuleStoryManager/ContainerStoryManager

get_story_nodes(*item*, *reverse=False*, *limit=False*)
Create a raw cypher query for story of an artifact and query neo4j with it.

Parameters

- **item** (*node*) – a Neo4j node whose story is requested by the user
- **reverse** (*bool*) – specifies the direction to proceed from current node corresponding to the story_flow
- **limit** (*bool*) – specifies if LIMIT keyword should be added to the created cypher query

Returns story paths for a particular artifact

Return type list

get_total_lead_time(*results*)
Get the total lead time - the time from the start of a story until its current state.

Parameters **results** (*list*) – contains inflated results from Neo4j

Returns the seconds of total time in the story, or None if sufficient data is not available

Return type int or None

get_total_processing_time(*results*)
Get the total time spent processing the story.

Parameters **results** (*list*) – contains inflated results from Neo4j

Returns the seconds of total time spent processing with a flag for inaccurate calculations

Return type tuple

get_wait_times(*results*)
Get the wait time between two artifacts for each pair of them, and the sum of these times.

Parameters **results** (*list*) – contains inflated results from Neo4j

Returns tuple with list of wait time ints in order of the story (oldest to newest), and a total wait time

Return type tuple

Raises **RuntimeError** – if results has less than 2 elements

set_story_labels(*requested_node_label*, *results*, *reverse=False*)
Replace Neo4j labels with appropriate labels of the story flow.

Parameters

- **requested_node_label** (*string*) – label of the node requested by the user

- **results** (*list*) – nodes in a story/path
- **reverse** (*bool*) – determines if the results are in reverse order of the story flow

Returns results with story/path labels

Return type list

story_flow(*label*)
Get the next/previous node in a story flow/pipeline path.

Parameters **label** (*str*) – Neo4j node label

Returns uid and relationship information in both forward and backward directions

Return type dict

class estuary.utils.story.**ContainerStoryManager**
A class containing utility methods to create a container story.

is_valid()
Determine if the story path matches the returned story.

Returns whether story is valid for this story path

Return type bool

story_flow(*label*)
Get the next/previous node in a story flow/pipeline path.

Parameters **label** (*str*) – Neo4j node label

Returns uid and relationship information in both forward and backward directions

Return type dict

class estuary.utils.story.**ModuleStoryManager**
A class containing utility methods to create a module story.

is_valid()
Determine if the story path matches the returned story.

Returns whether story is valid for this story path

Return type bool

story_flow(*label*)
Get the next/previous node in a story flow/pipeline path.

Parameters **label** (*str*) – Neo4j node label

Returns uid and relationship information in both forward and backward directions

Return type dict

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

e

s

Index

D

E

F

G

H

I

R

S

T

U

V