py42

Release py42 v1.11.1

Feb 02, 2021

Contents

1 Features 3

2 Content 5

Python Module Index 109

Index 111

license versions

`py42` is a Python wrapper around the Code42 REST APIs that also provides several utility methods. Use `py42` to develop your own tools for working with Code42 data while avoiding the overhead of session / authentication management.

CHAPTER 1

Features

- Managing users, organizations, and devices.
- Searching file events, alerts and auditlogs.
- Adding/Removing employees from detection lists.
- Managing cases.

CHAPTER 2

Content

2.1 User Guides

2.1.1 Getting started with py42

- *Licensing*
- *Installation*
- *Authentication*
- *Troubleshooting and Support*

Licensing

This project uses the MIT License.

Installation

You can install py42 from PyPI, from source, or from distribution.

From PyPI

The easiest and most common way is to use `pip`:

```
pip install py42
```

To install a previous version of py42 via `pip`, add the version number. For example, to install version 0.4.1, you would enter:

```
pip install py42==0.4.1
```

Visit the project history on PyPI to see all published versions.

From source

Alternatively, you can install py42 directly from source code:

```
git clone https://github.com/code42/py42.git
```

When it finishes downloading, from the root project directory, run:

```
python setup.py install
```

From distribution

If you want create a `.tar` ball for installing elsewhere, run this command from the project's root directory:

```
python setup.py sdist
```

After it finishes building, the `.tar` ball will be located in the newly created `dist` directory. To install it, enter:

```
pip install py42-[VERSION].tar.gz
```

Authentication

Important: py42 currently only supports token-based authentication.

To initialize the `py42.sdk.SDKClient`, you must provide your credentials (basic authentication). If you are writing a script, we recommend using a secure password storage library, such as `keyring`, for retrieving passwords. However, subsequent requests use JWT authentication.

If your account uses two-factor authentication, include the time-based one-time password (TOTP) when you initialize the `py42.sdk.SDKClient`. You can also provide a callable object that returns a TOTP. If you pass a callable, it will be called whenever a new TOTP is required to renew the authentication token.

py42 currently does **not** support SSO login providers or any other identity providers such as Active Directory or Okta.

Troubleshooting and support

Debug mode

Debug mode may be useful if you are trying to determine if you are experiencing permissions issues. When debug mode is on, py42 logs HTTP request data to the console's stderr. Use the following as a guide for how to turn on debug mode in py42:

```
import py42.sdk
import py42.settings
import logging

py42.settings.debug.level = logging.DEBUG
```

To provide your own logger, just replace `py42.settings.debug.logger`:

```
custom_logger = logging.getLogger("my_app")
handler = logging.FileHandler("my_app.log")
custom_logger.addHandler(handler)

py42.settings.debug.logger = custom_logger
```

File an issue on GitHub

If you are experiencing an issue with py42, you can create a *New issue* at the project repository. See the Github guide on creating an issue for more information.

Contact Code42 Support

If you don't have a GitHub account and are experiencing issues, contact Code42 support.

What's next?

Learn the basics by following the py42 Basics guide.

2.1.2 py42 Basics

This guide explains the basic concepts of py42. Learning these basics can help you gain confidence in writing your own scripts.

- *py42 Basics*
  - *Initialization*
  - *Paging*
  - *Py42Response*
  - *Dates*
  - *Exceptions*

The examples from this guide are intended as blanket concepts that apply to other areas in py42. For example, paging over users and devices works the same way as over departing employees and alerts.

Initialization

To use py42, you must initialize the SDK:

```
import py42.sdk

sdk = py42.sdk.from_local_account("https://console.us.code42.com", "my_username", "my_
↪password")
```

If your account uses two-factor authentication, include the time-based one-time password:

```
sdk = py42.sdk.from_local_account("https://console.us.code42.com", "my_username", "my_
↪password", totp="123456")
```

Alternatively, define a function that returns the time-based one-time password:

```
def promptForPassword():
    return input("Please input your authentication code: ")

sdk = py42.sdk.from_local_account("https://console.us.code42.com", "my_username", "my_
↪password", totp=promptForPassword)
```

Paging

py42 clients often have a method with the name (or name prefix) `get_all` which handles iterating over pages of response items. Here are some examples:

- `py42.sdk.devices.get_all()`
- `py42.sdk.users.get_all()`
- `py42.sdk.legalhold.get_all_matters()`
- `py42.sdk.orgs.get_all()`

These methods each return a python generator. Looping over the pages returned by the generator gives you access to the actual list of items. Use the code snippet below as an example for working with generators and paging in py42:

```
# Prints the username and notes for all departing employees

pages = sdk.detectionlists.departing_employee.get_all()  # pages has 'generator' type
for page in pages:  # page has 'Py42Response' type
    employees = page["items"]
    for employee in employees:
        username = employee["userName"]
        notes = employee["notes"]
        print("{0}: {1}".format(employee, notes))
```

Each page is a typical py42 response. The next section covers what you can do with `Py42Response` objects.

Py42Response

py42 clients return `Py42Response` objects which are intentionally similar to `requests.Response` objects. The `Py42Response` class hides unneeded metadata found on the raw `requests.Response.text` (which is available as `Py42Response.raw_text`), making it easier to get the most useful parts of the response. Also, the object is subscriptable, meaning you can access it with keys or indices (depending on the JSON type underneath `data` on Code42 API responses):

```
user = response["users"][0]
item = list_response[0]["itemProperty"]
```

To see all the keys on a response, observe its `.text` attribute. By printing the response, you essentially print its text property:

```
# Prints details about the response from a getting a detection list user.

response = sdk.detectionlists.get_user("test.user@example.com")
print(response)  # JSON as Dictionary - same as print(response.text)
print(response.raw_text)  # Raw API response
print(response.status_code)  # 200
```

(continues on next page)

(continued from previous page)

```
cloud_usernames = response["cloudUsernames"]
# if the response might not contain the property you're looking for,
# check to see if it exists with data.get
cloud_usernames = response.data.get("cloudUsernames")
if cloud_usernames:
    print(cloud_usernames)
```

Dates

Most dates in py42 support POSIX timestamps for date parameters. As an example, see :class:`sdk.queries.fileevents.filters.event_filter.EventTimestamp` which is used for querying file events by their event timestamp.

```
from datetime import datetime, timedelta

import py42.sdk
import py42.util
from py42.sdk.queries.fileevents.file_event_query import FileEventQuery
from py42.sdk.queries.fileevents.filters.event_filter import EventTimestamp

sdk = py42.sdk.from_local_account("https://console.us.code42.com", "my_username", "my_
↪password")

# Get the epoch date 14 days in the past
query_date = datetime.utcnow() - timedelta(days=14)
query_epoch = (query_date - datetime.utcfromtimestamp(0)).total_seconds()

query = FileEventQuery(EventTimestamp.on_or_after(query_epoch))

response = sdk.securitydata.search_file_events(query)

# Print all the md5 Checksums from every file event within the past 14 days.
file_events = response["fileEvents"]
for event in file_events:
    print(event["md5Checksum"])
```

Exceptions

py42 throws some of its own exceptions when failures occur. py42 exceptions are found in the `py42.sdk.exceptions` module. Some of the available exceptions are:

- `Py42ForbiddenError`: (403) With your currently signed-in account, you don't have the necessary permissions to perform the action you were trying to do.
- `Py42UnauthorizedError`: (401) The username or password is incorrect.
- `Py42InternalServerError`: (500) Likely an unhandled issue on our servers.

For example, you are making a `create_sdk()` function and want to print a more user-friendly message when the provided username or password are incorrect:

```
import keyring
import py42.sdk
from py42.exceptions import Py42UnauthorizedError

```

(continues on next page)

(continued from previous page)

```
def create_sdk(username):
    """Tries to initialize SDK. If unauthorized, prints message and exits."""
    try:
        password = keyring.get_password("my_program", username)
        return py42.sdk.from_local_account("www.authority.example.com", username,␣
↪password)
    except Py42UnauthorizedError:
        print("Invalid username or password.")
        exit(1)
```

2.1.3 Executing Searches

py42 features a powerful, flexible query system for quickly and easily searching file events and alerts. This guide explains the syntax for building queries and executing searches.

Search File Events

First, import the required modules and classes and create the SDK:

```
import py42.sdk
from py42.sdk.queries.fileevents.filters import *
from py42.sdk.queries.fileevents.file_event_query import FileEventQuery

sdk = py42.sdk.from_local_account("https://console.us.code42.com", "my_username", "my_
↪password")
```

You must create `query_filter.FilterGroup` objects to conduct searches. Filter groups have a type (in the form of a class), such as `EmailSender`, and an operator (in the form of a function), such as `is_in()`. Some example filter groups look like this:

```
email_filter = EmailSender.is_in(["test.user@example.com", "test.sender@example.com"])
exposure_filter = ExposureType.exists()
ip_filter = PrivateIPAddress.eq("127.0.0.1")
```

It is also possible to create `query_filter.FilterGroups` from raw JSON. For example:

```
raw_json = """{"filterClause":"AND","filters":[{"display":null,"value":"P1D","operator
↪":"WITHIN_THE_LAST","term":"eventTimestamp"}]}"""
json_dict = json.loads(raw_json)
filter_group = FilterGroup.from_dict(json_dict)
```

There are two operators when building `file_event_query.FileEventQuery` objects: `any()` and `all()`.

`any()` gets results where at least one of the filters is true and `all()` gets results where all of the filters are true.

```
any_query = FileEventQuery.any(email_filter, exposure_filter)
all_query = FileEventQuery.all(exposure_filter, ip_filter)
```

For convenience, the `FileEventQuery` constructor works the same way as `all()`:

```
all_query = FileEventQuery(exposure_filter, ip_filter)
```

You can put filters in an iterable and unpack them (using the `*` operator) in a `FileEventQuery`. This is a common use case for programs that need to conditionally build up filters:

```
# Conditionally appends filters to a list for crafting a query

filter_list = []
if need_shared:
    filter_list.append(Shared.is_true())
elif need_actors:
    actor_filter = Actor.is_in(["foo@example.com", "baz@example.com"])
    filter_list.append(actor_filter)
# Notice the use of the '*' operator to unpack filter_list
query = FileEventQuery(*filter_list)
```

To execute the search, use `securitydata.SecurityModule.search_file_events()`:

```
# Prints the MD5 hashes of all the files that caused exposure events where files were␣
↪moved to an external drive.

query = FileEventQuery(ExposureType.eq(ExposureType.REMOVABLE_MEDIA))
response = sdk.securitydata.search_file_events(query)
file_events = response["fileEvents"]
for event in file_events:
    print(event["md5Checksum"])
```

If the number of events exceeds 10,000 against a query, use `securitydata.SecurityModule.search_all_file_events()`:

```

query = FileEventQuery(ExposureType.eq(ExposureType.REMOVABLE_MEDIA))
response = sdk.securitydata.search_all_file_events(query)
file_events = response["fileEvents"]
for event in file_events:
    print(event["md5Checksum"])
while response["nextPgToken"] is not None:
    response = sdk.securitydata.search_all_file_events(query, page_token=response[
↪"nextPgToken"])
    file_events = response["fileEvents"]
    for event in file_events:
        print(event["md5Checksum"])
```

Search Alerts

Alert searches work in a very similar way to file event searches.

To start, import the filters and query object:

```
from py42.sdk.queries.alerts.filters import *
from py42.sdk.queries.alerts.alert_query import AlertQuery

# Create a query for getting all open alerts with severity either 'High' or 'Medium'.

filters = [AlertState.eq(AlertState.OPEN), Severity.is_in([Severity.HIGH, Severity.
↪MEDIUM])]
query = AlertQuery(*filters)
```

To execute the search, use the `alerts.AlertClient.search()` method:

```
# Prints the actor property from each search result
response = sdk.alerts.search(query)
alerts = response["alerts"]
for alert in alerts:
    print(alert["actor"])
```

2.1.4 Add or Remove Users From the Departing Employees List

Use py42 to quickly and easily manage users on the Departing Employees list. This guide describes how to add users to and remove users from the Departing Employees list.

To add a user to the Departing Employees list, all you need to know is the user's Code42 user UID.

To get the user UID based on username:

```
user = sdk.users.get_by_username("username")
uid = user["users"][0]["userUid"]
```

`user_id` below refers to the user UID.

```
from py42.exceptions import Py42UserAlreadyAddedError

# Add the departing employee
try:
    response = sdk.detectionlists.departing_employee.add(user_id, departure_date)
except Py42UserAlreadyAddedError:
    print("The user is already on the Departing Employee list.")
```

Important: If the user is already in the Departing Employees list, you will get an *py42.exceptions.Py42UserAlreadyAddedError*.

If a detection list user profile doesn't exist yet for this user, one will automatically be created before adding the user to the Departing Employees list.

To remove a user from the Departing Employees list:

```
sdk.detectionlists.departing_employee.remove(user_id)
```

For complete details, see Departing Employee.

2.1.5 High Risk Employee

Add or Remove Users From the High Risk Employee List

Use py42 to quickly and easily manage users on the High Risk Employee list. This guide describes how to add users to and remove users from the High Risk Employee list.

To add a user to the High Risk Employees list, all you need to know is the user's Code42 user UID.

To get the user UID based on username:

```
user = sdk.users.get_by_username("username")
uid = user["users"][0]["userUid"]
```

`user_id` below refers to the user UID.

```
# Add the high risk employee
response = sdk.detectionlists.high_risk_employee.add(user_id)
```

Important: If the user is already in the High Risk Employee list, you will get a response indicating that it is a bad request.

If a detection list user profile doesn't exist yet for this user, one will automatically be created before adding the user to the High Risk Employee list.

To remove a user from the High Risk Employee list:

```
sdk.detectionlists.high_risk_employee.remove(user_id)
```

For complete details, see High Risk Employee.

Add or Remove Risk Factors From Users

You can add/remove risk factor tags from a user programmatically using the `add_user_risk_tags()` and `remove_user_risk_tags()` methods in the `detectionlists` module. Both methods take a user_id and a list of tags that you want to add/remove:

```
tag_list = ["CONTRACT_EMPLOYEE", "ELEVATED_ACCESS_PRIVILEGES"]

# Add the risk tags
response = sdk.detectionlists.add_user_risk_tags(user_id, tag_list)

# Remove the risk tags
response = sdk.detectionlists.remove_user_risk_tags(user_id, tag_list)
```

The available risk tags are:

- `HIGH_IMPACT_EMPLOYEE`
- `ELEVATED_ACCESS_PRIVILEGES`
- `PERFORMANCE_CONCERNS`
- `FLIGHT_RISK`
- `SUSPICIOUS_SYSTEM_ACTIVITY`
- `POOR_SECURITY_PRACTICES`
- `CONTRACT_EMPLOYEE`

2.1.6 Get Active Devices From an Organization

Using py42, you can retrieve information about the active devices in your organization for various use cases. For example, you might want to create a simple report that illustrates how many devices are running each operating system in your Code42 environment. Your user role determines which devices you have access to.

To begin, initialize the SDK:

```
import py42.sdk
sdk = py42.sdk.from_local_account("https://console.us.code42.com", "my_username", "my_
↪password")
```

The `DeviceClient.get_all()` Function

Next, use `py42.sdk.clients.devices.DeviceClient` to search for active devices in your organization. Use the `active` parameter on the `get_all()` method.

The `active` parameter has three different states:

- If `active` is set to `True`, you will only get active devices.
- If `active` is set to `False`, you will only get deactivated devices.
- If you don't use `active`, you will get all devices.

The `get_all()` function returns a generator of pages of devices. The devices returned by `get_all()` are based on the role and permissions of the user authenticating the SDK.

Examples

Here is an example using `get_all()` to get all active devices in your organization(s):

```
# For each active device in your organization, print its GUID and operating system

response = sdk.devices.get_all(active=True)
for page in response:
    devices = page["computers"]
    for device in devices:
        print("{0} - {1}".format(device["guid"], device["osName"]))
```

As another example, you might have the Cross Org Administrator role and want to get all the active devices for just one of your organizations. To do this, use the `py42.sdk.clients.devices.OrgClient.get_by_name()` method. The `get_by_name()` method returns a list of organizations matching the name you give it.

```
# For each active device in the engineering organization, print its GUID and␣
↪operating system.

# Assume there is only one org named "Engineering"
engineering_org = sdk.orgs.get_by_name("Engineering")[0]
engineering_org_uid = engineering_org["orgUid"]
response = sdk.devices.get_all(active=True, org_uid=engineering_org_uid)
for page in response:
    devices = page["computers"]
    for device in devices:
        print("{0} - {1}".format(device["guid"], device["osName"]))
```

We got the org UID from the engineering organization and then passed it as a parameter to the method to get all the devices, thus getting all the active devices in the engineering organization.

2.1.7 View or Modify device settings

Use py42 to easily view and update the settings for devices with the `DeviceSettings` object.

The `DeviceSettings` object is a wrapper around the complex nested dict that the Code42 `Computer` API endpoint expects, providing helper properties that can be used to get/set values, without having to know the underlying nested structure.

To get started, create a `DeviceSettings` object for a given device guid:

```
device_settings = sdk.devices.get_settings(908765043021)
```

Some common non-modifiable details about the device are accessible as read-only properties:

```
>>> device_settings.computer_id
12345
>>> device_settings.guid
908765043021
>>> device_settings.org_id
42
>>> device_settings.user_id
494842
>>> device_settings.version
1525200006800
>>> device_settings.available_destinations
{'632540230984925185': 'PROe Cloud, US - West', '43': 'PROe Cloud, US'}
```

And to change settings, in most cases you can just assign new values to the corresponding attribute:

```
>>> device_settings.name
"Admin's Computer"
>>> device_settings.name = "Bob's Laptop"
```

Because device backup settings are tied to a given "Backup Set", of which there could be more than one, the `DeviceSettings.backup_sets` property returns a list of `BackupSet` wrapper classes that help manage backup configuration settings.

```
>>> device_settings.backup_sets
[<BackupSet: id: 1, name: 'Primary - Backup Set'>, <BackupSet: id: 298010138, name:
↪'Secondary (large files) - Backup Set'>]
```

See the Configuring Backup Sets guide for details on managing backup set settings.

For convenience and logging purposes, all changes are tracked in the `.changes` property of the `DeviceSettings` objects.

```
>>> device_settings.changes
{'destinations': "{'43': 'PROe Cloud, US <LOCKED>'} -> {'43': 'PROe Cloud, US <LOCKED>
↪', '632540230984925185': 'PROe Cloud, US - West'}"}
```

Once you've made all the desired changes to a `DeviceSettings` object, you can post the changes by passing it to the `sdk.devices.update_settings` method, which returns a `Py42Response` object with the server response:

```
>>> sdk.devices.update_settings(device_settings)
<Py42Response [status=200, data={'active': True, 'address': '192.168.74.144:4247',
↪'alertState': 0, 'alertStates': ['OK'], ...}]>
```

Advanced Usage

Because `DeviceSettings` is a subclass of `UserDict` with added attributes/methods to help easily access/modify setting values, the underlying dict that ultimately gets posted to the server is stored on the `.data` attribute of `DeviceSettings` instances, and a `DeviceSettings` object otherwise behaves like a normal dict.

If there is a setting that is not yet implemented by py42 as a helper method/attribute, those values can be manually managed by treating the `DeviceSettings` object as a normal dict.

For example, setting the "backup status email frequency" value to only send every 10 days, via the helper attribute:

```
>>> device_settings.backup_status_email_frequency_days = 10
```

And doing the same thing by setting the value manually on the underlying dict:

```
>>> device_settings["settings"]["serviceBackupConfig"]["backupStatusEmailFreqInMinutes
↪"] = "14400"
```

The benefits of the py42 helper attributes/methods is that the values mimic what the Console UI uses (in this case days vs the minutes expected by the API endpoint), so you don't have to worry about doing conversions yourself. But since the underlying dict is accessible, you aren't constrained to only what py42 has so far implemented.

Warning: When manually changing values on the underlying dict, those aren't registered in the *.changes* property and thus won't be captured in debug logs by the *sdk.devices.update_settings()* method.

2.1.8 View or Modify organization settings

Use py42 to easily view and update the settings for organizations with the `OrgSettings` object.

The `OrgSettings` object is a wrapper around the complex dicts that the Code42 `Org` and `OrgSettings` API endpoints expect, providing helper properties that can be used to get/set values, without having to know the underlying complexity of the APIs.

To get started, create a `OrgSettings` object for a given org_id:

```
org_settings = sdk.orgs.get_settings(org_id)
```

Some common non-modifiable details about the org are accessible as read-only properties:

```
>>> org_settings.org_id
424345
>>> org_settings.registration_key
'XXXX-YYYY-AAAA-BBBB'
```

And to change settings, in most cases you can just assign new values to the appropriate attribute:

```
>>> org_settings.name
'Admin Test Org'
>>> org_settings.name = "Admin Production Org"
```

Configuring device backup defaults for an org is very similar to configuring backup settings for an individual device, the `OrgSetting` object has a `.device_defaults` property that contains a `DeviceSettingsDefaults` object providing convenience attributes/methods for configuring defaults for all devices in the org.

```
>>> org_settings.device_defaults.backup_status_email_enabled
True
>>> org_settings.device_defaults.warning_alert_days
7
>>> org_settings.device_defaults.warning_alert_days = 14
```

Backup set configurations are contained in the `.device_defaults.backup_sets` property, and return a list of `BackupSet` wrapper classes for each set configured for the org:

```
>>> org_setting.device_defaults.backup_sets
[<BackupSet: id: 1, name: 'Production Environment - Backup Set'>]
```

See the Configuring Backup Sets guide for details on managing backup set settings.

Once you've made all the desired changes to an `OrgSettings` object, you can post the changes by passing it to the `sdk.orgs.update_settings()` method.

Because there are two endpoints that manage different organization settings values (`/api/Org` and `/api/OrgSettings`), the `sdk.orgs.update_settings()` method might make up to two requests to the server, depending on what `OrgSetting` values were actually modified. Because of the potential for two response values, `orgs.update_settings()` returns a `OrgSettingsResponse` namedtuple with the responses from both endpoints (if applicable), along with an `error` flag that indicates if any errors occurred. If an error occurred, the `org_response` or `org_settings_response` attributes will contain the `Py42Exception` that was raised instead of the `Py42Response`.

```
>>> sdk.orgs.update_settings(org_settings)
OrgSettingsResponse(error=False, org_response=<Py42Response [status=200, data={'active
↪': True, 'blocked': False, 'classification': 'BASIC', 'configInheritanceCounts': {},
↪ ...}]>, org_settings_response=None)
```

2.1.9 Configuring Backup Sets

Code42 devices' backup configurations are managed by "Backup Sets", which can be configured either at the individual device level, or set as default configurations at the org level.

The py42 `BackupSet` class can be used to view and change the settings of a given backup set.

`BackupSet` instances are automatically constructed by py42 and attached to their corresponding `DeviceSettings` or `OrgSettings` objects, and stored in the `.backup_sets` properties (`DeviceSettings.backup_sets` or `OrgSettings.device_defaults.backup_sets`).

The following examples will use an individual device's backup set, but all the methods/attributes are the same when configuring an org device default backup set.

Create a `DeviceSettings` object and get the primary backup set object:

```
>>> device_settings
>>> device_settings.backup_sets
[<BackupSet: id: 1, name: 'Primary - Backup Set'>, <BackupSet: id: 298010138, name:
↪'Secondary (large files) - Backup Set'>]
>>> bs = device_settings.backup_sets[0]
```

View/update destinations:

```
>>> bs.destinations
{'43': 'PROe Cloud, US <LOCKED>'}
>>>
```

(continues on next page)

(continued from previous page)

```
>>> bs.add_destination(587738803578339329)
>>> bs.remove_destination(43)
>>> bs.destinations
{'632540230984925185': 'PROe Cloud, US - West'}
```

View/update backup file selection/exclusion lists:

```
>>> bs.included_files
['C:/Users/Bob/']
>>> bs.excluded_files
[]
>>>
>>> bs.included_files.append("D:/")
>>> bs.excluded_files.append("C:/Users/Bob/Downloads")
```

You can also replace the existing list with a new one:

```
>>> bs.included_files = ["C:/Users/", "D:/"]
```

View/update filename exclusion patterns:

```
>>> bs.filename_exclusions
['.*/Photos/']
>>> bs.filename_exclusions.append(".*/Pictures/")
```

2.2 Method Documentation

The main SDK object by which all other methods are accessed is created by calling `py42.sdk.from_local_account`. For example:

```
import py42.sdk

sdk = py42.sdk.from_local_account("console.us.code42.com", "john.doe@example.com",
↪"my_pw")
# access properties on 'sdk' to explore all the available methods
```

Important: *py42* cannot be used with SAML or Sigle Sign-On based accounts such as Okta or Active Directory. Only accounts that are added by having an administrator create them within the Code42 console are currently supported.

Explore the complete public documentation for `py42` below.

2.2.1 Orgs

class py42.services.orgs.**OrgService**(*connection*)

Bases: py42.services.BaseService

A service for interacting with Code42 organization APIs.

Use the OrgService to create and retrieve organizations. You can also use it to block and deactivate organizations.

block(*org_id*)
Blocks the organization with the given org ID as well as its child organizations. A blocked organization will not allow any of its users or devices to log in. New registrations will be rejected and all currently logged in clients will be logged out. Backups continue for any devices that are still active. Rest Documentation

Parameters **org_id** (*int*) – An ID for an organization.

Returns *py42.response.Py42Response*

create_org(*org_name*, *org_ext_ref=None*, *notes=None*, *parent_org_uid=None*)
Creates a new organization. REST Documentation

Parameters

- **org_name** (*str*) – The name of the new organization.
- **org_ext_ref** (*str, optional*) – External reference information, such as a serial number, asset tag, employee ID, or help desk issue ID. Defaults to None.
- **notes** (*str, optional*) – Descriptive information about the organization. Defaults to None.
- **parent_org_uid** (*int, optional*) – The org UID for the parent organization. Defaults to None.

Returns *py42.response.Py42Response*

deactivate(*org_id*)
Deactivates the organization with the given ID, including all users, plans, and devices. Backups stop and archives move to cold storage. REST Documentation

Parameters **org_id** (*int*) – An ID for an organization.

Returns *py42.response.Py42Response*

get_agent_full_disk_access_states(*org_id*)
Gets the full disk access status for devices in an org. REST Documentation

Parameters **org_id** (*str*) – The org's identifier.

Returns A response containing settings information.

Return type *py42.response.Py42Response*

get_agent_state(*org_id*, *property_name*)
Gets the agent state of the devices in the org. REST Documentation

Parameters

- **org_id** (*str*) – The org's identifier.
- **property_name** (*str*) – The name of the property to retrieve (e.g. *fullDiskAccess*).

Returns A response containing settings information.

Return type *py42.response.Py42Response*

get_all(***kwargs*)
Gets all organizations. REST Documentation

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of organizations.

Return type generator

get_by_id(*org_id*, ***kwargs*)
Gets the organization with the given ID. REST Documentation

Parameters **org_id** (*int*) – An ID for an organization.

Returns A response containing the organization.

Return type *py42.response.Py42Response*

get_by_uid(*org_uid*, ***kwargs*)
Gets the organization with the given UID. REST Documentation

Parameters **org_uid** (*str*) – A UID for an organization.

Returns A response containing the organization.

Return type *py42.response.Py42Response*

get_current(***kwargs*)
Gets the organization for the currently signed-in user. REST Documentation

Returns A response containing the organization for the currently signed-in user.

Return type *py42.response.Py42Response*

get_page(*page_num*, *page_size=None*, ***kwargs*)
Gets an individual page of organizations. REST Documentation

Parameters

- **page_num** (*int*) – The page number to request.
- **page_size** (*int, optional*) – The number of organizations to return per page. Defaults to *py42.settings.items_per_page*.
- **kwargs** (*dict, optional*) – Additional advanced-user arguments. Defaults to None.

Returns *py42.response.Py42Response*

get_settings(*org_id*)
Gets setting data for an org and returns an *OrgSettingsManager* for the target org.

Parameters **org_id** (*int,str*) – The identifier of the org.

Returns A class to help manage org settings.

Return type py42.clients._settings_managers.OrgSettings

reactivate(*org_id*)
Reactivates the organization with the given ID. Backups are *not* restarted automatically. REST Documentation

Parameters **org_id** (*int*) – An ID for an organization.

Returns *py42.response.Py42Response*

unblock(*org_id*)
Removes a block, if one exists, on an organization and its descendants with the given ID. All users in the organization remain blocked until they are unblocked individually. REST Documentation

Parameters **org_id** (*int*) – An ID for an organization.

Returns *py42.response.Py42Response*

update_settings(*org_settings*)
Updates an org's settings based on changes to the passed in *OrgSettings* instance.

Parameters **org_settings** (*OrgSettings*) – An *OrgSettings* instance with desired modifications to settings.

Returns A namedtuple containing the result of the setting change api calls.

Return type py42.services.orgs.OrgSettings

2.2.2 Org Settings

class py42.clients.settings.org_settings.**OrgSettings**(*org_settings*, *t_settings*)
Bases: collections.UserDict, object

Class used to manage an Organization's settings.

archive_hold_days
Number of days backup archives are held in cold storage after deactivation or destination removal from any devices in this Org.

backup_alert_recipient_emails
List of email addresses that organization backup alert emails get sent to (org admin users get these automatically).

backup_critical_email_days
The number of days devices in this org can go without any backup before "critical" alerts get sent to org admins.

backup_warning_email_days
The number of days devices in this org can go without any backup before "warning" alerts get sent to org admins.

endpoint_monitoring_background_priority_enabled
Determines if devices in this org have reduced priority in some IO bound tasks. If enabled, devices may see improved general device performance at the expense of some Code42 backup/security tasks taking longer.

endpoint_monitoring_browser_and_applications_enabled
Determines if browser and other application activity endpoint monitoring event capturing is enabled for this org.

endpoint_monitoring_cloud_sync_enabled
Determines if cloud sync endpoint monitoring event capturing is enabled for this org.

endpoint_monitoring_custom_applications_mac
List of additional applications the Code42 client monitors for file exfiltration activity.

See Support Documentation for more details.

endpoint_monitoring_custom_applications_win
List of additional applications the Code42 client monitors for file exfiltration activity.

See Support Documentation for more details.

endpoint_monitoring_enabled
Determines if endpoint monitoring settings are enabled for this org.

Disabling this property also disables "removable media", "cloud sync", "browser and application monitoring" and "printer detection" properties.

endpoint_monitoring_file_exfiltration_detection_exclusions
File types and file paths to exclude from file exfiltration detection.

See Support Documentation for more details on the shape of the body this setting expects.

endpoint_monitoring_file_metadata_collection_enabled
Determines if file metadata collection is enabled for this org.

endpoint_monitoring_file_metadata_collection_exclusions
File types and file paths to exclude from file metadata collection.

See Support Documentation for more details on the shape of the body this setting expects.

endpoint_monitoring_file_metadata_ingest_scan_enabled
Determines if file metadata collection does an initial full scan when first enabled on devices.

endpoint_monitoring_file_metadata_scan_enabled
Determines if file metadata collection regular full scans are enabled for this org.

endpoint_monitoring_printer_detection_enabled
Determines if printer endpoint monitoring event capturing is enabled for this org.

endpoint_monitoring_removable_media_enabled
Determines if removable media endpoint monitoring event capturing is enabled for this org.

external_reference
External reference field for this Org.

maximum_user_subscriptions
Number of users allowed to consume a license in this Org. Set to -1 for unlimited.

notes
Notes field for this Org.

org_backup_quota
Backup storage quota (in GB) for this organization. Set to -1 for unlimited.

org_id
The identifier for the org.

org_name
Name for this Org.

packets
The setting packets for any modifications to be posted to the /api/OrgSettings endpoint.

quota_settings_inherited
Determines if Org Quota settings (*maximum_user_subscriptions*, *org_backup_quota*, *user_backup_quota*, *archive_hold_days*) are inherited from parent organization.

Modifying one of the Org Quota attributes automatically sets this attribute to *False*.

registration_key
The registration key for the org.

reporting_settings_inherited
Determines if Org Reporting settings (*backup_warning_email_days*, *backup_critical_email_days', 'backup_alert_recipient_emails*) are inherited from parent organization.

Modifying one of the Org Reporting attributes automatically sets this attribute to *False*.

user_backup_quota
Backup storage quota (in GB) for each user in this organization. Set to -1 for unlimited.

web_restore_admin_limit
Limit (in MB) to amount of data restorable by admin users via web restore.

web_restore_enabled
Determines if web restores are enabled for devices in this org.

web_restore_user_limit
Limit (in MB) to amount of data restorable by non-admin users via web restore.

class py42.clients.settings.device_settings.**DeviceSettingsDefaults**(*device_dict*, *org_settings*)

Bases: collections.UserDict, object

Class used for managing an Organization's Device Default settings. Also acts as a base class for *DeviceSettings* to manage individual device settings.

available_destinations
Returns a dict of destinations available to be used by devices. Dict keys are destination guids and values are destination names.

backup_status_email_enabled
Determines if the regularly scheduled backup status email is enabled.

backup_status_email_frequency_days
Determines the frequency of the regularly scheduled backup status email.

critical_alert_days
The number of days a device can go without any backup activity before "warning" alert threshold is passed.

critical_email_enabled
Determines if backup "critical" threshold email alerts are configured for this device.

warning_alert_days
The number of days a device can go without any backup activity before "warning" alert threshold is passed.

warning_email_enabled
Determines if backup "warning" threshold email alerts are configured for this device.

2.2.3 Users

class py42.services.users.**UserService**(*connection*)

Bases: py42.services.BaseService

A service for interacting with Code42 user APIs. Use the UserService to create and retrieve users. You can also use it to block and deactivate users.

add_role(*user_id*, *role_name*)
Adds a role to a user. REST Documentation

Parameters

- **user_id** (*int*) – An ID for a user.
- **role_name** (*str*) – The name of the role to assign to the user.

Returns *py42.response.Py42Response*

block(*user_id*)
Blocks the user with the given ID. A blocked user is not allowed to log in or restore files. Backups will continue if the user is still active. REST Documentation

Parameters **user_id** (*int*) – An ID for a user.

Returns *py42.response.Py42Response*

change_org_assignment(*user_id*, *org_id*)
Assigns a user to a different organization. REST Documentation

Parameters

- **user_id** (*int*) – An ID for a user.
- **org_id** (*int*) – An ID for the organization to move the user to.

Returns *py42.response.Py42Response*

create_user(*org_uid*, *username*, *email*, *password=None*, *first_name=None*, *last_name=None*, *notes=None*)

Creates a new user. WARNING: If the provided username already exists for a user, it will be updated in the database instead. REST Documentation

Parameters

- **org_uid** (*str*) – The org UID for the organization the new user belongs to.
- **username** (*str*) – The username for the new user.
- **email** (*str*) – The email for the new user.
- **password** (*str, optional*) – The password for the new user. Defaults to None.
- **first_name** (*str, optional*) – The first name for the new user. Defaults to None.
- **last_name** (*str, optional*) – The last name for the new user. Defaults to None.
- **notes** (*str, optional*) – Descriptive information about the user. Defaults to None.

Returns *py42.response.Py42Response*

deactivate(*user_id*, *block_user=None*)

Deactivates the user with the given user ID. Backups discontinue for a deactivated user, and their archives go to cold storage. REST Documentation

Parameters

- **user_id** (*int*) – An ID for a user.
- **block_user** (*bool, optional*) – Blocks the user upon deactivation. Defaults to None.

Returns *py42.response.Py42Response*

get_all(*active=None*, *email=None*, *org_uid=None*, *role_id=None*, *q=None*, ***kwargs*)

Gets all users. REST Documentation

Parameters

- **active** (*bool, optional*) – True gets active users only, and false gets deactivated users only. Defaults to None.
- **email** (*str, optional*) – Limits users to only those with this email. Defaults to None.
- **org_uid** (*str, optional*) – Limits users to only those in the organization with this org UID. Defaults to None.
- **role_id** (*int, optional*) – Limits users to only those with a given role ID. Defaults to None.
- **q** (*str, optional*) – A generic query filter that searches across name, username, and email. Defaults to None.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of users.

Return type generator

get_available_roles()

Report the list of roles that are available for the authenticated user to assign to other users. V4 REST Documentation

Returns *py42.response.Py42Response*

get_by_id(*user_id*, ***kwargs*)
Gets the user with the given ID. REST Documentation

Parameters **user_id** (*int*) – An ID for a user.

Returns A response containing the user.

Return type *py42.response.Py42Response*

get_by_uid(*user_uid*, ***kwargs*)
Gets the user with the given UID. REST Documentation

Parameters **user_uid** (*str*) – A UID for a user.

Returns A response containing the user.

Return type *py42.response.Py42Response*

get_by_username(*username*, ***kwargs*)
Gets the user with the given username. REST Documentation

Parameters **username** (*str or unicode*) – A username for a user.

Returns A response containing the user.

Return type *py42.response.Py42Response*

get_current(***kwargs*)
Gets the currently signed in user. REST Documentation

Returns A response containing the user.

Return type *py42.response.Py42Response*

get_page(*page_num*, *active=None*, *email=None*, *org_uid=None*, *role_id=None*, *page_size=None*, *q=None*, ***kwargs*)
Gets an individual page of users. REST Documentation

Parameters

- **page_num** (*int*) – The page number to request.
- **active** (*bool, optional*) – True gets active users only, and false gets deactivated users only. Defaults to None.
- **email** (*str, optional*) – Limits users to only those with this email. Defaults to None.
- **org_uid** (*str, optional*) – Limits users to only those in the organization with this org UID. Defaults to None.
- **role_id** (*int, optional*) – Limits users to only those with a given role ID. Defaults to None.
- **page_size** (*int, optional*) – The number of items on the page. Defaults to *py42.settings.items_per_page*.
- **q** (*str, optional*) – A generic query filter that searches across name, username, and email. Defaults to None.

Returns *py42.response.Py42Response*

get_roles(*user_id*)
Return the list of roles that are currently assigned to the given user. REST Documentation

Parameters **user_id** (*int*) – An ID for a user.

Returns *py42.response.Py42Response*

get_scim_data_by_uid(*user_uid*)
Returns SCIM data such as division, department, and title for a given user. REST Documentation

Parameters **user_uid** (*str*) – A Code42 user uid.

Returns *py42.response.Py42Response*

reactivate(*user_id*, *unblock_user=None*)
Reactivates the user with the given ID. REST Documentation

Parameters

- **user_id** (*int*) – An ID for a user.
- **unblock_user** (*bool, optional*) – Whether or not to unblock the user. Defaults to None.

Returns *py42.response.Py42Response*

remove_role(*user_id*, *role_name*)
Removes a role from a user. REST Documentation <https://console.us.code42.com/apidocviewer/#UserRole-delete>‘__

Parameters

- **user_id** (*int*) – An ID for a user.
- **role_name** (*str*) – The name of the role to unassign from the user.

Returns *py42.response.Py42Response*

unblock(*user_id*)
Removes a block, if one exists, on the user with the given user ID. Unblocked users are allowed to log in and restore. REST Documentation

Parameters **user_id** (*int*) – An ID for a user.

Returns *py42.response.Py42Response*

class py42.usercontext.**UserContext**(*administration_client*)
Bases: object

An object representing the currently logged in user.

get_current_tenant_id()
Gets the currently signed in user’s tenant ID.

2.2.4 Devices

class py42.services.devices.**DeviceService**(*connection*)
Bases: py42.services.BaseService

A class to interact with Code42 device/computer APIs.

block(*device_id*)
Blocks a device causing the user not to be able to log in to or restore from Code42 on that device. REST Documentation

Parameters **device_id** (*int*) – The identification number of the device.

Returns *py42.response.Py42Response*

deactivate(*device_id*)
Deactivates a device, causing backups to stop and archives to go to cold storage. REST Documentation

Parameters **device_id** (*int*) – The identification number of the device.

Returns *py42.response.Py42Response*

deauthorize(*device_id*)
Deauthorizes the device with the given ID. If used on a cloud connector device, it will remove the authorization token for that account. REST Documentation

Parameters **device_id** (*int*) – The identification number of the device.

Returns *py42.response.Py42Response*

get_agent_full_disk_access_state(*guid*)
Gets the full disk access status of a device. REST Documentation

Parameters **guid** (*str*) – The globally unique identifier of the device.

Returns A response containing settings information.

Return type *py42.response.Py42Response*

get_agent_state(*guid*, *property_name*)
Gets the agent state of the device. REST Documentation

Parameters

- **guid** (*str*) – The globally unique identifier of the device.
- **property_name** (*str*) – The name of the property to retrieve (e.g. *fullDiskAccess*).

Returns A response containing settings information.

Return type *py42.response.Py42Response*

get_all(*active=None*, *blocked=None*, *org_uid=None*, *user_uid=None*, *destination_guid=None*, *include_backup_usage=None*, *include_counts=True*, *q=None*, ***kwargs*)
Gets all device information.

When no arguments are passed, all records are returned. To filter results, specify respective arguments. For example, to retrieve all active and blocked devices, pass active=true and blocked=true. REST Documentation

Parameters

- **active** (*bool, optional*) – Filters results by device state. When set to True, gets all active devices. When set to False, gets all deactivated devices. When set to None or excluded, gets all devices regardless of state. Defaults to None.
- **blocked** (*bool, optional*) – Filters results by blocked status: True or False. Defaults to None.
- **org_uid** (*int, optional*) – The identification number of an Organization. Defaults to None.
- **user_uid** (*int, optional*) – The identification number of a User. Defaults to None.
- **destination_guid** (*str or int, optional*) – The globally unique identifier of the storage server that the device back up to. Defaults to None.
- **include_backup_usage** (*bool, optional*) – A flag to denote whether to include the destination and its backup stats. Defaults to None.
- **include_counts** (*bool, optional*) – A flag to denote whether to include total, warning, and critical counts. Defaults to True.

- **q** (*str, optional*) – Searches results flexibly by incomplete GUID, hostname, computer name, etc. Defaults to None.

Returns

An object that iterates over *py42.response.Py42Response* objects that each contain a page of devices.

The devices returned by *get_all()* are based on the role and permissions of the user authenticating the py42 SDK.

Return type generator

get_by_guid(*guid*, *include_backup_usage=None*, ***kwargs*)
Gets device information by GUID. REST Documentation

Parameters

- **guid** (*str*) – The globally unique identifier of the device.
- **include_backup_usage** (*bool, optional*) – A flag to denote whether to include the destination and its backup stats. Defaults to None.

Returns A response containing device information.

Return type *py42.response.Py42Response*

get_by_id(*device_id*, *include_backup_usage=None*, ***kwargs*)
Gets device information by ID. REST Documentation

Parameters

- **device_id** (*int*) – The identification number of the device.
- **include_backup_usage** (*bool, optional*) – A flag to denote whether to include the destination and its backup stats. Defaults to None.

Returns A response containing device information.

Return type *py42.response.Py42Response*

get_page(*page_num*, *active=None*, *blocked=None*, *org_uid=None*, *user_uid=None*, *destination_guid=None*, *include_backup_usage=None*, *include_counts=True*, *page_size=None*, *q=None*)
Gets a page of devices. REST Documentation

Parameters

- **page_num** (*int*) – The page number to request.
- **active** (*bool, optional*) – Filters results by device state. When set to True, gets all active devices. When set to False, gets all deactivated devices. When set to None or excluded, gets all devices regardless of state. Defaults to None.
- **blocked** (*bool, optional*) – Filters results by blocked status: True or False. Defaults to None.
- **org_uid** (*int, optional*) – The identification number of an Organization. Defaults to None.
- **user_uid** (*int, optional*) – The identification number of a User. Defaults to None.
- **destination_guid** (*str or int, optional*) – The globally unique identifier of the storage server that the device back up to. Defaults to None.
- **include_backup_usage** (*bool, optional*) – A flag to denote whether to include the destination and its backup stats. Defaults to None.

- **include_counts** (*bool, optional*) – A flag to denote whether to include total, warning, and critical counts. Defaults to True.
- **page_size** (*int, optional*) – The number of devices to return per page. Defaults to *py42.settings.items_per_page*.
- **q** (*str, optional*) – Searches results flexibly by incomplete GUID, hostname, computer name, etc. Defaults to None.

Returns *py42.response.Py42Response*

get_settings (*guid*)
Gets setting data for a device and returns a *DeviceSettings* object for the target device.

Parameters **guid** (*int, str*) – The globally unique identifier of the device.

Returns A class to help manage device settings.

Return type *py42.clients.settings.device_settings.DeviceSettings*

reactivate (*device_id*)
Activates a previously deactivated device. REST Documentation

Parameters **device_id** (*int*) – The identification number of the device.

Returns *py42.response.Py42Response*

unblock (*device_id*)
Unblocks a device, permitting a user to be able to login and restore again. REST Documentation

Parameters **device_id** (*int*) – The identification number of the device.

Returns *py42.response.Py42Response*

update_settings (*device_settings*)
Updates a device's settings based on changes to the passed in *DeviceSettings* instance.

Parameters **device_settings** (*DeviceSettings*) – An instance of *DeviceSettings* with desired modifications to settings.

Returns A response containing the result of the setting change.

Return type *py42.response.Py42Response*

2.2.5 Device Settings

class py42.clients.settings.device_settings.**DeviceSettings** (*device_dict*)
Bases: *py42.clients.settings.device_settings.DeviceSettingsDefaults*

Class used to manage an individual device's settings.

backup_sets = None
List of *BackupSet* objects used to manage this device's backup set configurations.

computer_id
Identifier of this device. Read-only.

device_id
Identifier of this device (alias of *.computer_id*). Read only.

external_reference
External reference field for this device.

guid
Globally unique identifier of this device. Read-only.

java_memory_heap_max
The maximum memory the client will use on its system

name
Name for this device.

notes
Notes field for this device.

org_id
Identifier of the organization this device belongs to. Read-only.

user_id
Identifier of the user this device belongs to. Read-only.

version
Latest reported Code42 client version number for this device. Read-only.

2.2.6 Backup Sets

class py42.clients.settings.device_settings.**BackupSet**(*settings_manager*, *backup_set_dict*)

Bases: collections.UserDict, object

Helper class for managing device backup sets and Org device default backup sets.

add_destination(*destination_guid*)
Adds a destination to be used by this backup set. Raises a Py42Error if the supplied destination guid is not available to the parent device/org.

Parameters **destination_guid** (*str, int*) – The globally unique identifier of the destination to be added.

destinations
Returns a dict of the destinations used for backup for the backup set. Dict keys are the destination guids, values are the destination names.

excluded_files
Returns the list of files/folders excluded from the backup selection. Items can be added/removed from this list via normal list methods, or assigning a new list of files to this attribute to replace the existing one.

filename_exclusions
Returns the list of regex patterns used to exclude file paths from the backup selection. Items can be added/removed from this list via normal list methods, or assigning a new list of patterns to this attribute to replace the existing one.

included_files
Returns the list of files/folders included in the backup selection. Items can be added/removed from this list via normal list methods, or assigning a new list of files to this attribute to replace the existing one.

lock_destination(*destination_guid*)
Locks an in-use destination, disallowing the device owner from removing this destination from their backup. Raises a Py42Error if the supplied destination guid is not in use on this backup set, or not available to the parent device/org.

locked
Indicates whether the backup set as a whole is locked. If True, individual settings for this backup set (except for Destination settings), cannot be modified.

remove_destination(*destination_guid*)
Removes a destination from use by this backup set.

Parameters **destination_guid** (*str, int*) – The globally unique identifier of the destination to be removed.

unlock_destination(*destination_guid*)
Unlocks an in-use destination, allowing the device owner to remove this destination from their backup. Raises a `Py42Error` if the supplied destination guid is not in use on this backup set, or not available to the parent device/org.

2.2.7 Security Data

class py42.clients.securitydata.**SecurityDataClient**(*security_service*, *file_event_service*, *preservation_data_service*, *saved_search_service*, *storage_service_factory*)

Bases: `object`

get_all_plan_security_events(*plan_storage_info*, *cursor=None*, *include_files=True*, *event_types=None*, *min_timestamp=None*, *max_timestamp=None*)
Gets events for legacy Endpoint Monitoring file activity on removable media, in cloud sync folders, and browser uploads. Support Article

Parameters

- **plan_storage_info** (*py42.clients.securitydata.PlanStorageInfo*) – Information about storage nodes for a plan to get file event activity for.
- **cursor** (*str, optional*) – A cursor position for only getting file events you did not previously get. Defaults to None.
- **include_files** (*bool, optional*) – Whether to include the files related to the file events.
- **to None.** (*Defaults*) –
- **event_types** – (str, optional): A comma-separated list of event types to filter by.

  Available options are:

  - DEVICE_APPEARED
  - DEVICE_DISAPPEARED
  - DEVICE_FILE_ACTIVITY
  - PERSONAL_CLOUD_FILE_ACTIVITY
  - RESTORE_JOB
  - RESTORE_FILE
  - FILE_OPENED
  - RULE_MATCH
  - DEVICE_SCAN_RESULT
  - PERSONAL_CLOUD_SCAN_RESULT

  Defaults to None.

- **min_timestamp** (*int or float or str or datetime, optional*) – Timestamp in milliseconds or str format "yyyy-MM-DD HH:MM:SS" or a datetime instance. Defaults to None.
- **max_timestamp** (*int or float or str or datetime, optional*) – Timestamp in milliseconds or str format "yyyy-MM-DD HH:MM:SS" or a datetime instance. Defaults to None.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of events.

Return type generator

get_all_user_security_events(*user_uid*, *cursor=None*, *include_files=True*, *event_types=None*, *min_timestamp=None*, *max_timestamp=None*)

Gets legacy Endpoint Monitoring file activity events for the user with the given UID.

Parameters

- **user_uid** (*str*) – The UID of the user to get security events for.
- **cursor** (*str, optional*) – A cursor position for only getting events you did not previously get. Defaults to None.
- **include_files** (*bool, optional*) – Whether to include the files related to the file activity events. Defaults to None.
- **event_types** – (str, optional): A comma-separated list of event types to filter by.

  Available options are:

  - DEVICE_APPEARED
  - DEVICE_DISAPPEARED
  - DEVICE_FILE_ACTIVITY
  - PERSONAL_CLOUD_FILE_ACTIVITY
  - RESTORE_JOB
  - RESTORE_FILE
  - FILE_OPENED
  - RULE_MATCH
  - DEVICE_SCAN_RESULT
  - PERSONAL_CLOUD_SCAN_RESULT

  Defaults to None.
- **min_timestamp** (*int or float or str or datetime, optional*) – Timestamp in milliseconds or str format "yyyy-MM-DD HH:MM:SS" or a datetime instance. Defaults to None.
- **max_timestamp** (*int or float or str or datetime, optional*) – Timestamp in milliseconds or str format "yyyy-MM-DD HH:MM:SS" or a datetime instance. Defaults to None.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of events.

Return type generator

get_security_plan_storage_info_list(*user_uid*)
Gets IDs (plan UID, node GUID, and destination GUID) for the storage nodes containing the file activity event data for the user with the given UID. REST Documentation

Parameters **user_uid** (*str*) – The UID of the user to get plan storage information for.

Returns list[*py42.clients.securitydata.PlanStorageInfo*]

savedsearches
A collection of methods related to retrieving forensic search data.

Returns class: *py42._internal.services.securitydata.SavedSearchService*

search_all_file_events(*query*, *page_token=''*)
Searches for all file events, returning a page of events with a token in the response to retrieve next page.

REST Documentation

Parameters

- **query** (str or *py42.sdk.queries.fileevents.file_event_query.FileEventQuery*) – The file event query to filter search results.
- **page_token** (*str, optional*) – A token used to indicate the starting point for additional page results. For the first page, do not pass `page_token`. For all consecutive pages, pass the token from the previous response from field `nextPgToken`. When using `page_token`, any sorting parameters from the *FileEventQuery* will be ignored. Defaults to empty string.

Returns A response containing page of events.

Return type *py42.response.Py42Response*

search_file_events(*query*)
Searches for file events, returns up to the first 10,000 events. REST Documentation

Parameters **query** (str or *py42.sdk.queries.fileevents.file_event_query.FileEventQuery*) – The file event query to filter search results.

Returns A response containing the first 10,000 events.

Return type *py42.response.Py42Response*

stream_file_by_md5(*checksum*)
Stream file based on MD5 checksum.

Parameters **checksum** (*str*) – MD5 hash of the file.

Returns Returns a stream of the requested file.

stream_file_by_sha256(*checksum*)
Stream file based on SHA256 checksum.

Parameters **checksum** (*str*) – SHA256 hash of the file.

Returns Returns a stream of the requested file.

class py42.clients.securitydata.**PlanStorageInfo**(*plan_uid*, *destination_guid*, *node_guid*)
Bases: `object`

destination_guid
The GUID of the destination containing the storage archive.

node_guid
The GUID of the storage node containing the archive.

plan_uid
The UID of the storage plan.

2.2.8 Legal Hold

class py42.services.legalhold.**LegalHoldService**(*connection*)
Bases: py42.services.BaseService

A service for interacting with Code42 Legal Hold APIs.

The LegalHoldService provides the ability to manage Code42 Legal Hold Policies and Matters. It can: - Create, view, and list all existing Policies. - Create, view, deactivate, reactivate, and list all existing Matters. - Add/remove Custodians from a Matter.

add_to_matter(*user_uid*, *legal_hold_uid*)
Add a user (Custodian) to a Legal Hold Matter. REST Documentation

Parameters

- **user_uid** (*str*) – The identifier of the user.
- **legal_hold_uid** (*str*) – The identifier of the Legal Hold Matter.

Returns *py42.response.Py42Response*

create_matter(*name*, *hold_policy_uid*, *description=None*, *notes=None*, *hold_ext_ref=None*)
Creates a new, active Legal Hold Matter. REST Documentation

Parameters

- **name** (*str*) – The name of the new Legal Hold Matter.
- **hold_policy_uid** (*str*) – The identifier of the Preservation Policy that will apply to this Matter.
- **description** (*str, optional*) – An optional description of the Matter. Defaults to None.
- **notes** (*str, optional*) – Optional notes information. Defaults to None.
- **hold_ext_ref** (*str, optional*) – Optional external reference information. Defaults to None.

Returns *py42.response.Py42Response*

create_policy(*name*, *policy=None*)
Creates a new Legal Hold Preservation Policy. V4 REST Documentation

Parameters

- **name** (*str*) – The name of the new Policy.
- **policy** (*dict, optional*) – The desired Preservation Policy settings as a dict. Defaults to None (where the server-default backup set is used).

Returns *py42.response.Py42Response*

deactivate_matter(*legal_hold_uid*)
Deactivates and closes a Legal Hold Matter. V4 REST Documentation

Parameters **legal_hold_uid** (*str*) – The identifier of the Legal Hold Matter.

Returns *py42.response.Py42Response*

get_all_matter_custodians(*legal_hold_uid=None*, *user_uid=None*, *user=None*, *active=True*)

Gets all Legal Hold memberships.

Each user (Custodian) who has been added to a Legal Hold Matter is returned by the server as a LegalHoldMembership object in the response body. If the object's active state is "INACTIVE", they have been removed from the Matter and are no longer subject to the Legal Hold retention rules. Users can be Custodians of multiple Legal Holds at once (and thus would be part of multiple LegalHoldMembership objects). REST Documentation

Parameters

- **legal_hold_uid** (*str, optional*) – Find LegalHoldMemberships for the Legal Hold Matter with this unique identifier. Defaults to None.
- **user_uid** (*str, optional*) – Find LegalHoldMemberships for the user with this identifier. Defaults to None.
- **user** (*str, optional*) – Find LegalHoldMemberships by flexibly searching on username, email, extUserRef, or last name. Will find partial matches. Defaults to None.
- **active** (*bool or None, optional*) – Find LegalHoldMemberships by their active state. True returns active LegalHoldMemberships, False returns inactive LegalHoldMemberships, None returns all LegalHoldMemberships regardless of state. Defaults to True.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of LegalHoldMembership objects.

Return type generator

get_all_matters(*creator_user_uid=None*, *active=True*, *name=None*, *hold_ext_ref=None*)

Gets all existing Legal Hold Matters. REST Documentation

Parameters

- **creator_user_uid** (*str, optional*) – Find Matters by the identifier of the user who created them. Defaults to None.
- **active** (*bool or None, optional*) – Find Matters by their active state. True returns active Matters, False returns inactive Matters, None returns all Matters regardless of state. Defaults to True.
- **name** (*str, optional*) – Find Matters with a 'name' that either equals or contains this value. Defaults to None.
- **hold_ext_ref** (*str, optional*) – Find Matters having a matching external reference field. Defaults to None.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of Legal Hold Matters.

Return type generator

get_custodians_page(*page_num*, *legal_hold_membership_uid=None*, *legal_hold_uid=None*, *user_uid=None*, *user=None*, *active=True*, *page_size=None*)

Gets an individual page of Legal Hold memberships. One of the following optional args is required to determine which custodians to retrieve:

legal_hold_membership_uid, *legal_hold_uid*, *user_uid*, *user*

REST Documentation

Parameters

- **page_num** (*int*) – The page number to request.
- **legal_hold_membership_uid** (*str, optional*) – Find LegalHoldMemberships with a specific membership UID. Defaults to None.
- **legal_hold_uid** (*str, optional*) – Find LegalHoldMemberships for the Legal Hold Matter with this unique identifier. Defaults to None.
- **user_uid** (*str, optional*) – Find LegalHoldMemberships for the user with this identifier. Defaults to None.
- **user** (*str, optional*) – Find LegalHoldMemberships by flexibly searching on username, email, extUserRef, or last name. Will find partial matches. Defaults to None.
- **active** (*bool or None, optional*) – Find LegalHoldMemberships by their active state. True returns active LegalHoldMemberships, False returns inactive LegalHoldMemberships, None returns all LegalHoldMemberships regardless of state. Defaults to True.
- **page_size** (*int, optional*) – The size of the page. Defaults to *py42.settings.items_per_page*.

Returns

Return type *py42.response.Py42Response*

get_matter_by_uid(*legal_hold_uid*)

Gets a single Legal Hold Matter. REST Documentation

Parameters **legal_hold_uid** (*str*) – The identifier of the Legal Hold Matter.

Returns A response containing the Matter.

Return type *py42.response.Py42Response*

get_matters_page(*page_num*, *creator_user_uid=None*, *active=True*, *name=None*, *hold_ext_ref=None*, *page_size=None*)

Gets a page of existing Legal Hold Matters. REST Documentation

Parameters

- **page_num** (*int*) – The page number to request.
- **creator_user_uid** (*str, optional*) – Find Matters by the identifier of the user who created them. Defaults to None.
- **active** (*bool or None, optional*) – Find Matters by their active state. True returns active Matters, False returns inactive Matters, None returns all Matters regardless of state. Defaults to True.
- **name** (*str, optional*) – Find Matters with a 'name' that either equals or contains this value. Defaults to None.
- **hold_ext_ref** (*str, optional*) – Find Matters having a matching external reference field. Defaults to None.
- **page_size** (*int, optional*) – The number of legal hold items to return per page. Defaults to *py42.settings.items_per_page*.

Returns

Return type *py42.response.Py42Response*

get_policy_by_uid(*legal_hold_policy_uid*)
Gets a single Preservation Policy. V4 REST Documentation

Parameters **legal_hold_policy_uid** (*str*) – The identifier of the Preservation Policy.

Returns A response containing the Policy.

Return type *py42.response.Py42Response*

get_policy_list()
Gets a list of existing Preservation Policies. V4 REST Documentation

Returns A response containing the list of Policies.

Return type *py42.response.Py42Response*

reactivate_matter(*legal_hold_uid*)
Reactivates and re-opens a closed Matter. REST Documentation

Parameters **legal_hold_uid** (*str*) – The identifier of the Legal Hold Matter.

Returns *py42.response.Py42Response*

remove_from_matter(*legal_hold_membership_uid*)
Remove a user (Custodian) from a Legal Hold Matter. REST Documentation

Parameters **legal_hold_membership_uid** (*str*) – The identifier of the LegalHoldMembership representing the Custodian to Matter relationship.

Returns *py42.response.Py42Response*

2.2.9 Detection Lists

class py42.clients.detectionlists.**DetectionListsClient**(*user_profile_service*, *departing_employee_service*, *high_risk_employee_service*)

Bases: object

add_user_cloud_alias(*user_id*, *alias*)
Add a cloud alias to a user.

Parameters

- **user_id** (*str or int*) – The userUid of the user whose alias you want to update.
- **alias** (*str*) – The alias to be added.

Returns *py42.response.Py42Response*

add_user_risk_tags(*user_id*, *tags*)
Add one or more risk factor tags.

Parameters

- **user_id** (*str or int*) – The userUid of the user whose risk factor tag(s) you want to update.
- **tags** (*str or list of str*) – A single tag or multiple tags in a list to be added. For example: `"tag1"` or `["tag1", "tag2"]`. For python version 2.X, pass `u"str"` instead of `"str"`.

Returns *py42.response.Py42Response*

create_user(*username*)
Create a detection list profile for a user.

Parameters **username** (*str*) – The Code42 username of the user.

Returns *py42.response.Py42Response*

get_user(*username*)
Get user details by username.

Parameters **username** (*str*) – The Code42 username of the user.

Returns *py42.response.Py42Response*

get_user_by_id(*user_id*)
Get user details by user_id.

Parameters **user_id** (*str or int*) – The Code42 userId of the user.

Returns *py42.response.Py42Response*

refresh_user_scim_attributes(*user_id*)
Refresh SCIM attributes of a user. REST documentation

Parameters **user_id** (*str or int*) – The userUid of the user whose attributes you wish to refresh.

Returns *py42.response.Py42Response*

remove_user_cloud_alias(*user_id*, *alias*)
Remove a cloud alias from a user.

Parameters

- **user_id** (*str or int*) – The userUid of the user whose alias needs to be removed.
- **alias** (*str*) – The alias to be removed.

Returns *py42.response.Py42Response*

remove_user_risk_tags(*user_id*, *tags*)
Remove one or more risk factor tags.

Parameters

- **user_id** (*str or int*) – The userUid of the user whose risk factor tag(s) needs you want to remove.
- **tags** (*str or list of str*) – A single tag or multiple tags in a list to be removed. For example: `"tag1"` or `["tag1", "tag2"]`. For python version 2.X, pass `u"str"` instead of `"str"`.

Returns *py42.response.Py42Response*

update_user_notes(*user_id*, *notes*)
Add or update notes related to the user.

Parameters

- **user_id** (*str or int*) – The userUid of the user whose notes you want to update.
- **notes** (*str*) – User profile notes.

Returns *py42.response.Py42Response*

Departing Employees

class py42.services.detectionlists.departing_employee.**DepartingEmployeeFilters**
Bases: py42.services.detectionlists._DetectionListFilters

Constants available for filtering Departing Employee search results.

class py42.services.detectionlists.departing_employee.**DepartingEmployeeService**(*session*, *user_context*, *user_profile_serv*

Bases: py42.services.BaseService

A service for interacting with Code42 Departing Employee APIs.

add(*user_id*, *departure_date=None*)
Adds a user to the Departing Employees list. Creates a detection list user profile if one didn't already exist. REST Documentation

Raises a Py42BadRequestError when a user already exists in the Departing Employee detection list.

Parameters

- **user_id** (*str or int*) – The Code42 userUid of the user you want to add to the departing employees list.
- **departure_date** (*str or datetime, optional*) – Date in yyyy-MM-dd format or instance of datetime. Date is treated as UTC. Defaults to None.

Returns *py42.response.Py42Response*

get(*user_id*)
Gets departing employee data of a user. REST Documentation

Parameters **user_id** (*str or int*) – The Code42 userUid of the user.

Returns *py42.response.Py42Response*

get_all(*filter_type='OPEN'*, *sort_key='CREATED_AT'*, *sort_direction='DESC'*, *page_size=100*)
Gets all Departing Employees.

Parameters

- **filter_type** (*str, optional*) – EXFILTRATION_30_DAYS, EXFILTRATION_24_HOURS, OPEN, or LEAVING_TODAY. Constants are available at *py42.services.detectionlists.departing_employee.DepartingEmployeeFilters*. Defaults to "OPEN".
- **sort_key** (*str, optional*) – Sort results based by field. Defaults to "CREATED_AT".
- **sort_direction** (*str, optional*) – ASC or DESC. Defaults to "DESC".
- **page_size** (*int, optional*) – The number of departing employees to return per page. Defaults to 100.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of departing employees.

Return type generator

get_page(*page_num*, *filter_type='OPEN'*, *sort_key='CREATED_AT'*, *sort_direction='DESC'*, *page_size=100*)
Gets a single page of Departing Employees.

Parameters

- **page_num** (*int*) – The page number to request.
- **filter_type** (*str, optional*) – EXFILTRATION_30_DAYS, EXFILTRATION_24_HOURS, OPEN, or LEAVING_TODAY. Constants are available at *py42.services.detectionlists.departing_employee.DepartingEmployeeFilters*. Defaults to "OPEN".
- **sort_key** (*str, optional*) – Sort results based by field. Defaults to "CREATED_AT".
- **sort_direction** (*str. optional*) – ASC or DESC. Defaults to "DESC".
- **page_size** (*int, optional*) – The number of departing employees to return per page. Defaults to 100.

Returns *py42.response.Py42Response*

remove(*user_id*)
Removes a user from the Departing Employees list. REST Documentation

Parameters **user_id** (*str or int*) – The Code42 userUid of the user.

Returns *py42.response.Py42Response*

set_alerts_enabled(*alerts_enabled=True*)
Enable or disable email alerting on Departing Employee exposure events. REST Documentation

Parameters **alerts_enabled** (*bool*) – Set alerting to on (True) or off (False). Defaults to True.

Returns *py42.response.Py42Response*

update_departure_date(*user_id*, *departure_date*)
Add or modify details of an existing Departing Employee case. REST Documentation

Parameters

- **user_id** (*str*) – The Code42 userUid of the user.
- **departure_date** (*str or datetime*) – Date in yyyy-MM-dd format or instance of datetime. Date is treated as UTC.

Returns *py42.response.Py42Response*

High Risk Employee

class py42.services.detectionlists.high_risk_employee.**HighRiskEmployeeFilters**
Bases: py42.services.detectionlists._DetectionListFilters

Constants available for filtering High Risk Employee search results.

class py42.services.detectionlists.high_risk_employee.**HighRiskEmployeeService**(*connection*, *user_context*, *user_profile_servic*

Bases: py42.services.BaseService

A service for interacting with High Risk Employee APIs.

add(*user_id*)
Adds a user to the High Risk Employee detection list. Creates a detection list user profile if one didn't already exist.

Raises a Py42BadRequestError when a user already exists in the High Risk Employee detection list.

Parameters **user_id** (*str or int*) – The Code42 userUid of the user you want to add to the High Risk Employee detection list.

Returns *py42.response.Py42Response*

get (*user_id*)
Gets user information.

Parameters **user_id** (*str or int*) – The Code42 userUid of the user has been added to the High Risk Employee detection list.

Returns *py42.response.Py42Response*

get_all (*filter_type='OPEN'*, *sort_key=None*, *sort_direction=None*, *page_size=100*)
Searches High Risk Employee list. Filter results by filter_type.

Parameters

- **filter_type** (*str, optional*) – EXFILTRATION_30_DAYS, EXFILTRATION_24_HOURS, or OPEN. Constants are available at *py42.services.detectionlists.high_risk_employee.HighRiskEmployeeFilters*. Defaults to "OPEN".
- **sort_key** (*str, optional*) – Sort results based by field. Defaults to None.
- **sort_direction** (*str, optional*) – ASC or DESC. Constants available at *py42.constants.SortDirection*. Defaults to None.
- **page_size** (*int, optional*) – The number of high risk employees to return per page. Defaults to 100.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of users.

Return type generator

get_page (*page_num*, *filter_type='OPEN'*, *sort_key=None*, *sort_direction=None*, *page_size=100*)
Gets a single page of High Risk Employees.

Parameters

- **page_num** (*int*) – The page number to request.
- **filter_type** (*str, optional*) – EXFILTRATION_30_DAYS, EXFILTRATION_24_HOURS, or OPEN. Constants are available at *py42.services.detectionlists.high_risk_employee.HighRiskEmployeeFilters*. Defaults to "OPEN".
- **sort_key** (*str, optional*) – Sort results based by field. Defaults to None.
- **sort_direction** (*str. optional*) – ASC or DESC. Constants available at *py42.constants.SortDirection*. Defaults to None.
- **page_size** (*int, optional*) – The number of high risk employees to return per page. Defaults to 100.

Returns *py42.response.Py42Response*

remove (*user_id*)
Removes a user from the High Risk Employee detection list.

Parameters **user_id** (*str or int*) – The Code42 userUid of the user you want to add to the High Risk Employee detection list.

Returns *py42.response.Py42Response*

set_alerts_enabled(*enabled=True*)
Enables alerts.

Parameters **enabled** (*bool*) – Whether to enable alerts for all users.

Returns *py42.response.Py42Response*

2.2.10 Filter Classes

The following classes construct filters for file event queries. Each filter class corresponds to an alert detail. Call the appropriate classmethod on your desired filter class with the `value` you want to match and it will return a `FilterGroup` object that can be passed to `AlertQuery`'s `all()` or `any()` methods to create complex queries that match multiple filter rules.

See *Executing Searches* for more on building search queries.

class py42.sdk.queries.alerts.filters.alert_filter.**Actor**
Bases: *py42.sdk.queries.alerts.filters.alert_filter.AlertQueryFilterStringField*

Class that filters alerts based on the username that originated the event(s) that triggered the alert.

classmethod contains(*value*)
Creates a *FilterGroup* for filtering results where the value with key `self._term` contains the given value. Useful for creating `CONTAINS` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters **value** (*str*) – The value used to match on.

Returns *FilterGroup*

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_contains(*value*)
Creates a *FilterGroup* for filtering results where the value with key `self._term` does not contain the given value. Useful for creating `DOES_NOT_CONTAIN` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters **value** (*str*) – The value used to exclude on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.alerts.filters.alert_filter.**AlertQueryFilterStringField**
Bases: *py42.sdk.queries.query_filter.QueryFilterStringField*

classmethod contains(*value*)
Creates a *FilterGroup* for filtering results where the value with key `self._term` contains the given value. Useful for creating `CONTAINS` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters **value** (*str*) – The value used to match on.

Returns *FilterGroup*

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_contains(*value*)
Creates a *FilterGroup* for filtering results where the value with key `self._term` does not contain the given value. Useful for creating `DOES_NOT_CONTAIN` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters **value** (*str*) – The value used to exclude on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.alerts.filters.alert_filter.**AlertState**
Bases: *py42.sdk.queries.query_filter.QueryFilterStringField*

Class that filters alerts based on alert state.

Available options are:

- `AlertState.OPEN`
- `AlertState.DISMISSED`
- `AlertState.PENDING`
- `AlertState.IN_PROGRESS`

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.alerts.filters.alert_filter.**DateObserved**
Bases: *py42.sdk.queries.query_filter.QueryFilterTimestampField*

Class that filters alerts based on the timestamp the alert was triggered.

classmethod in_range(*start_value*, *end_value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is in range between the provided `start_value` and `end_value`.

Parameters

- **start_value** (*str or int or float or datetime*) – The start value used to filter results.
- **end_value** (*str or int or float or datetime*) – The end value used to filter results.

Returns *FilterGroup*

classmethod on_or_after(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is on or after the provided ``value`.

Parameters **value** (*str or int or float or datetime*) – The value used to filter results.

Returns *FilterGroup*

classmethod on_or_before(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is on or before the provided `value`.

Parameters **value** (*str or int or float or datetime*) – The value used to filter results.

Returns *FilterGroup*

classmethod on_same_day(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is within the same calendar day as the provided `value`.

Parameters **value** (*str or int or float or datetime*) – The value used to filter results.

Returns *FilterGroup*

class py42.sdk.queries.alerts.filters.alert_filter.**Description**
Bases: *py42.sdk.queries.alerts.filters.alert_filter.AlertQueryFilterStringField*

Class that filters alerts based on rule description text.

classmethod contains(*value*)
Creates a *FilterGroup* for filtering results where the value with key `self._term` contains the given value. Useful for creating `CONTAINS` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters **value** (*str*) – The value used to match on.

Returns *FilterGroup*

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_contains(*value*)
Creates a *FilterGroup* for filtering results where the value with key `self._term` does not contain the given value. Useful for creating `DOES_NOT_CONTAIN` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters **value** (*str*) – The value used to exclude on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.alerts.filters.alert_filter.**RuleId**
Bases: *py42.sdk.queries.query_filter.QueryFilterStringField*

Class that filters alerts based on rule identifier.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.alerts.filters.alert_filter.**RuleName**
Bases: *py42.sdk.queries.alerts.filters.alert_filter.AlertQueryFilterStringField*

Class that filters alerts based on rule name.

classmethod contains(*value*)
Creates a *FilterGroup* for filtering results where the value with key `self._term` contains the given value. Useful for creating `CONTAINS` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters **value** (*str*) – The value used to match on.

Returns *FilterGroup*

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_contains(*value*)
Creates a *FilterGroup* for filtering results where the value with key `self._term` does not contain the given value. Useful for creating `DOES_NOT_CONTAIN` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters **value** (*str*) – The value used to exclude on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.alerts.filters.alert_filter.**RuleSource**
Bases: *py42.sdk.queries.query_filter.QueryFilterStringField*

Class that filters alerts based on rule source.

Available options are:

- `RuleSource.ALERTING`
- `RuleSource.DEPARTING_EMPLOYEE`
- `RuleSource.HIGH_RISK_EMPLOYEE`

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.alerts.filters.alert_filter.**RuleType**
Bases: *py42.sdk.queries.query_filter.QueryFilterStringField*

Class that filters alerts based on rule type.

Available options are:

- `RuleType.ENDPOINT_EXFILTRATION`
- `RuleType.CLOUD_SHARE_PERMISSIONS`
- `RuleType.FILE_TYPE_MISMATCH`

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.alerts.filters.alert_filter.**Severity**
Bases: *py42.sdk.queries.query_filter.QueryFilterStringField*

Class that filters alerts based on severity.

Available options are:

- `Severity.HIGH`
- `Severity.MEDIUM`
- `Severity.LOW`

classmethod eq(*value*)

Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)

Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)

Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)

Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

py42.sdk.queries.alerts.filters.alert_filter.**create_contains_filter_group**(*term*, *value*)

Creates a *FilterGroup* for filtering results where the value with key `term` contains the given value. Useful for creating `CONTAINS` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters

- **term** – (str): The term of the filter, such as `actor`.
- **value** (*str*) – The value used to match on.

Returns *FilterGroup*

py42.sdk.queries.alerts.filters.alert_filter.**create_not_contains_filter_group**(*term*, *value*)

Creates a *FilterGroup* for filtering results where the value with key `term` does not contain the given value. Useful for creating `DOES_NOT_CONTAIN` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters

- **term** – (str): The term of the filter, such as `actor`.
- **value** (*str*) – The value used to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.alerts.alert_query.**AlertQuery**(**args*, ***kwargs*)
Bases: py42.sdk.queries.BaseQuery

Helper class for building Code42 Alert queries.

An AlertQuery instance's `all()` and `any()` take one or more *FilterGroup* objects to construct a query that can be passed to the `AlertService.search()` method. `all()` returns results that match all of the provided filter criteria, `any()` will return results that match any of the filters.

For convenience, the *AlertQuery* constructor does the same as `all()`.

Usage example:

```
state_filter = AlertState.eq(AlertState.OPEN)
rule_name_filter = RuleName.contains("EmailRule")
query = AlertQuery.all(state_filter, rule_name_filter)
```

2.2.11 Alerts

class py42.clients.alerts.**AlertsClient**(*alert_service*, *alert_rules_client*)
Bases: object

get_details(*alert_ids*)
Gets the details for the alerts with the given IDs, including the file event query that, when passed into a search, would result in events that could have triggered the alerts.

Parameters **alert_ids** (*iter[str]*) – The identification numbers of the alerts for which you want to get details for.

Returns A response containing the alert details.

Return type *py42.response.Py42Response*

reopen(*alert_ids*, *reason=None*)
Reopens the resolved alerts with the given IDs.

Parameters

- **alert_ids** (*iter[str]*) – The identification numbers for the alerts to reopen.
- **reason** (*str, optional*) – The reason the alerts are reopened. Defaults to None.

Returns *py42.response.Py42Response*

resolve(*alert_ids*, *reason=None*)
Resolves the alerts with the given IDs.

Parameters

- **alert_ids** (*iter[str]*) – The identification numbers for the alerts to resolve.
- **reason** (*str, optional*) – The reason the alerts are now resolved. Defaults to None.

Returns *py42.response.Py42Response*

rules
A collection of methods for managing alert rules.

Returns py42.services.alertrules.AlertRulesClient

search(*query*)
Searches alerts using the given *py42.sdk.queries.alerts.alert_query.AlertQuery*.

Parameters **query** (*py42.sdk.queries.alerts.alert_query.AlertQuery*) – An alert query. See the *Executing Searches User Guide* to learn more about how to construct a query.

Returns A response containing the alerts that match the given query.

Return type *py42.response.Py42Response*

update_state(*status*, *alert_ids*, *note=None*)
Update status for given alert IDs.

Parameters

- **status** (*str*) – Status to set from OPEN, RESOLVED, PENDING, IN_PROGRESS
- **alert_ids** (*iter[str]*) – The identification numbers for the alerts to reopen.
- **note** (*str, optional*) – User note regarding the status. Defaults to None.

Returns *py42.response.Py42Response*

2.2.12 Alert Rules

class py42.clients.alertrules.**AlertRulesClient**(*alerts_service*, *alert_rules_service*)
Bases: object

add_user(*rule_id*, *user_id*)
Update alert rule to monitor user aliases against the Uid for the given rule id.

Parameters

- **rule_id** (*str*) – Observer Id of a rule to be updated.
- **user_id** (*str*) – The Code42 userUid of the user to add to the alert

Returns *py42.response.Py42Response*

cloudshare
A collection of methods for managing cloud sharing alert rules.

Returns py42.services.alertrules.cloud_share.CloudShareService

exfiltration
A collection of methods for managing exfiltration alert rules.

Returns py42.services.alertrules.exfiltration.ExfiltrationService

filetypemismatch
A collection of methods for managing file type mismatch alert rules.

Returns py42.services.alertrules.file_type_mismatch.FileTypeMismatchService

get_all(*sort_key='CreatedAt'*, *sort_direction='DESC'*)
Fetch all available rules.

Parameters

- **sort_key** (*str, optional*) – Sort results based by field. Defaults to 'CreatedAt'.

- **sort_direction** (*str, optional*) – ASC or DESC. Constants available at *py42.constants.SortDirection*. Defaults to "DESC"

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of rules.

Return type generator

get_all_by_name(*rule_name*)
Search for matching rules by name.

Parameters **rule_name** (*str*) – Rule name to search for, case insensitive search.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of rules with the given name.

Return type generator

get_by_observer_id(*observer_id*)
Get the rule with the matching observer ID.

Parameters **observer_id** (*str*) – The observer ID of the rule to return.

Returns *py42.response.Py42Response*

get_page(*sort_key='CreatedAt'*, *sort_direction='DESC'*, *page_num=1*, *page_size=None*)
Gets a page of alert rules. Note that you can use page_size here the same way as other methods that have a *page_size* parameter in py42. However, under the hood, it subtracts one from the given page size in the implementation as the Code42 alerts API expected the start page to be zero while the rest of the Code42 APIs expect the start page to be one.

Parameters

- **sort_key** (*str, optional*) – Sort results based by field. Defaults to "CreatedAt".
- **sort_direction** (*str, optional*) – ASC or DESC. Constants available at *py42.constants.SortDirection*. Defaults to "DESC".
- **page_num** (*int, optional*) – The page number to get. Defaults to 1.
- **page_size** (*int, optional*) – The number of items per page. Defaults to *py42.settings.items_per_page*.

Returns *py42.response.Py42Response*

remove_all_users(*rule_id*)
Update alert rule criteria to remove all users the from the alert rule.

Parameters **rule_id** (*str*) – Observer rule Id of a rule to be updated.

Returns *py42.response.Py42Response*

remove_user(*rule_id*, *user_id*)
Update alert rule criteria to remove a user and all its aliases from a rule.

Parameters

- **rule_id** (*str*) – Observer rule Id of a rule to be updated.
- **user_id** (*str*) – The Code42 userUid of the user to remove from the alert

Returns *py42.response.Py42Response*

Exfiltration rules

class py42.services.alertrules.**ExfiltrationService**(*connection*, *tenant_id*)
Bases: py42.services.BaseService

get(*rule_id*)
Fetch exfiltration alert rule by rule id.

Parameters **rule_id** (*str*) – Observer rule Id of a rule to be fetched.

Returns *py42.response.Py42Response*

Cloud share rules

class py42.services.alertrules.**CloudShareService**(*connection*, *tenant_id*)
Bases: py42.services.BaseService

get(*rule_id*)
Fetch cloud share alert rule by rule id.

Parameters **rule_id** (*str*) – Observer rule Id of a rule to be fetched.

Returns *py42.response.Py42Response*

File type mismatch rules

class py42.services.alertrules.**FileTypeMismatchService**(*connection*, *tenant_id*)
Bases: py42.services.BaseService

get(*rule_id*)
Fetch File type mismatch alert rules by rule id.

Parameters **rule_id** (*str*) – Observer rule Id of a rule to be fetched.

Returns *py42.response.Py42Response*

2.2.13 Shared Query Filters

class py42.sdk.queries.query_filter.**FilterGroup**(*filter_list*, *filter_clause='AND'*)
Bases: object

Class for constructing a logical sub-group of related filters from a list of *QueryFilter* objects. Takes a list of *QueryFilter* objects and combines them logically using the passed in filter clause (AND or OR).

When str() is called on a *FilterGroup* instance, the combined filter items are transformed into a JSON string to be used as part of a Forensic Search or Alert query.

When dict() is called on a *FilterGroup* instance, the combined filter items are transformed into the Python *dict* equivalent of their JSON representation. This can be useful for programmatically manipulating a *FilterGroup* after it's been created.

filter_clause
The clause joining the filters, such as AND or OR.

filter_list
The list of *QueryFilter* objects in this group.

classmethod from_dict(*_dict*)
Creates an instance of *FilterGroup* from the values found in `_dict`. `_dict` must contain keys `filters` and `filterClause`.

Parameters **_dict** (*dict*) – A dictionary containing keys `term`, `operator`, and `value`.

Returns *FilterGroup*

class py42.sdk.queries.query_filter.**QueryFilter**(*term*, *operator*, *value=None*)
Bases: `object`

Class for constructing a single filter object for use in a search query.

When `str()` is called on a *QueryFilter* instance, the (`term`, `operator`, `value`) attribute combination is transformed into a JSON string to be used as part of a Forensic Search or Alert query.

When `dict()` is called on a *QueryFilter* instance, the (`term`, `operator`, `value`) attribute combination is transformed into the Python *dict* equivalent of their JSON representation. This can be useful for programmatically manipulating a *QueryFilter* after it's been created.

classmethod from_dict(*_dict*)
Creates an instance of *QueryFilter* from the values found in `_dict`. `_dict` must contain keys `term`, `operator`, and `value`.

Parameters **_dict** (*dict*) – A dictionary containing keys `term`, `operator`, and `value`.

Returns *QueryFilter*

operator
The operator between `term` and `value`, such as `IS` or *IS_NOT*.

term
The term of the filter, such as `actor` or `sharedWith`.

value
The value used to filter results.

class py42.sdk.queries.query_filter.**QueryFilterBooleanField**
Bases: `object`

Helper class for creating filters where the search value is a boolean.

classmethod is_false()
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is False.

Returns *FilterGroup*

classmethod is_true()
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is True.

Returns *FilterGroup*

class py42.sdk.queries.query_filter.**QueryFilterStringField**
Bases: `object`

Helper class for creating filters where the search value is a string.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.query_filter.**QueryFilterTimestampField**
Bases: `object`

Helper class for creating filters where the search value is a timestamp.

classmethod in_range(*start_value*, *end_value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is in range between the provided `start_value` and `end_value`.

Parameters

- **start_value** (*str or int or float or datetime*) – The start value used to filter results.
- **end_value** (*str or int or float or datetime*) – The end value used to filter results.

Returns *FilterGroup*

classmethod on_or_after(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is on or after the provided ``value`.

Parameters **value** (*str or int or float or datetime*) – The value used to filter results.

Returns *FilterGroup*

classmethod on_or_before(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is on or before the provided `value`.

Parameters **value** (*str or int or float or datetime*) – The value used to filter results.

Returns *FilterGroup*

classmethod on_same_day(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is within the same calendar day as the provided `value`.

Parameters **value** (*str or int or float or datetime*) – The value used to filter results.

Returns *FilterGroup*

py42.sdk.queries.query_filter.**create_eq_filter_group**(*term*, *value*)
"Creates a *FilterGroup* for filtering results where the value with key `term` equals the given value. Useful for creating `IS` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters

- **term** – (str): The term of the filter, such as `actor` or `sharedWith`.
- **value** (*str*) – The value used to match on.

Returns *FilterGroup*

py42.sdk.queries.query_filter.**create_filter_group**(*query_filter_list*, *filter_clause*)
Creates a *FilterGroup* object. Useful for programmatically crafting query filters, such as filters not yet defined in py42. Alternatively, if you want to create custom filter groups with already defined operators (such as *IS* or *IS_IN*), see the other methods in this module, such as *create_eq_filter_group()*.

Parameters

- **query_filter_list** (*list*) – a list of *QueryFilter* objects.
- **filter_clause** (*str*) – The clause joining the filters, such as `AND` or `OR`.

Returns *FilterGroup*

py42.sdk.queries.query_filter.**create_in_range_filter_group**(*term*, *start_value*, *end_value*)
"Creates a *FilterGroup* for filtering results where the value with key `term` is in the given range. Examples include values describing dates. Useful for creating a combination of `ON_OR_AFTER` and `ON_OR_BEFORE` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters

- **term** – (str): The term of the filter, such as `eventTimestamp`.
- **start_value** (*str or int*) – The start value used to filter results.
- **end_value** (*str or int*) – The end value used to filter results.

Returns *FilterGroup*

py42.sdk.queries.query_filter.**create_is_in_filter_group**(*term*, *value_list*)
"Creates a *FilterGroup* for filtering results where the value with key `term` is one of several values. Useful for creating `IS_IN` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters

- **term** – (str): The term of the filter, such as `actor` or `sharedWith`.
- **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

py42.sdk.queries.query_filter.**create_not_eq_filter_group**(*term*, *value*)
"Creates a *FilterGroup* for filtering results where the value with key `term` does not equal the given value. Useful for creating `IS_NOT` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters

- **term** – (str): The term of the filter, such as `actor` or `sharedWith`.
- **value** (*str*) – The value used to exclude on.

Returns *FilterGroup*

py42.sdk.queries.query_filter.**create_not_in_filter_group**(*term*, *value_list*)
"Creates a *FilterGroup* for filtering results where the value with key `term` is not one of several values. Useful for creating `NOT_IN` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters

- **term** – (str): The term of the filter, such as `actor` or `sharedWith`.
- **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

py42.sdk.queries.query_filter.**create_on_or_after_filter_group**(*term*, *value*)
"Creates a *FilterGroup* for filtering results where the value with key `term` is on or after the given value. Examples include values describing dates. Useful for creating `ON_OR_AFTER` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters

- **term** – (str): The term of the filter, such as `eventTimestamp`.
- **value** (*str or int*) – The value used to filter results.

Returns *FilterGroup*

py42.sdk.queries.query_filter.**create_on_or_before_filter_group**(*term*, *value*)
"Creates a *FilterGroup* for filtering results where the value with key `term` is on or before the given value. Examples include values describing dates. Useful for creating `ON_OR_BEFORE` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters

- **term** – (str): The term of the filter, such as `eventTimestamp`.
- **value** (*str or int*) – The value used to filter results.

Returns *FilterGroup*

py42.sdk.queries.query_filter.**create_query_filter**(*term*, *operator*, *value=None*)
Creates a *QueryFilter* object. Useful for programmatically crafting query filters, such as filters not yet defined in py42.

Parameters

- **term** (*str*) – The term of the filter, such as `actor` or `sharedWith`.
- **operator** (*str*) – The operator between `term` and `value`, such as `IS` or *IS_NOT*.
- **value** (*str*) – The value used to filter results.

Returns *QueryFilter*

py42.sdk.queries.query_filter.**create_within_the_last_filter_group**(*term*, *value*)
Returns a *FilterGroup* that is useful for finding results where the key `term` is an `EventTimestamp._term` and the value is one of the *EventTimestamp* attributes as *value*.

Parameters **value** (*str*) – *EventTimestamp* attribute.

Returns *FilterGroup*

2.2.14 File Event Queries

class py42.sdk.queries.fileevents.file_event_query.**FileEventQuery**(**args*, ***kwargs*)

Bases: py42.sdk.queries.BaseQuery

Helper class for building Code42 Forensic Search queries.

A FileEventQuery instance's `all()` and `any()` take one or more *FilterGroup* objects to construct a query that can be passed to the `FileEventService.search()` method. `all()` returns results that match all of the provided filter criteria, `any()` will return results that match any of the filters.

For convenience, the *FileEventQuery* constructor does the same as `all()`.

Usage example:

```
email_filter = EmailSender.is_in(["test.user@example.com", "test.sender@example.
↪com"])
exposure_filter = ExposureType.exists()
query = FileEventQuery.all(email_filter, exposure_filter)
```

Saved Searches

class py42.services.savedsearch.**SavedSearchService**(*connection*, *file_event_client*)

Bases: py42.services.BaseService

A service to interact with saved search APIs.

execute(*search_id*, *page_number=None*, *page_size=None*)

Execute a saved search for given search Id and return its results.

Parameters

- **search_id** (*str*) – Unique search Id of the saved search.
- **page_number** (*int, optional*) – The consecutive group of results of size page_size in the result set to return. Defaults to None.
- **page_size** (*int, optional*) – The maximum number of results to be returned. Defaults to None.

Returns *py42.response.Py42Response*

get()

Fetch details of existing saved searches.

Returns *py42.response.Py42Response*

get_by_id(*search_id*)

Fetch the details of a saved search by its given search Id.

Parameters **search_id** (*str*) – Unique search Id of the saved search.

Returns *py42.response.Py42Response*

get_query(*search_id*, *page_number=None*, *page_size=None*)

Get the saved search in form of a query(*py42.sdk.queries.fileevents.file_event_query*).

Parameters

- **search_id** (*str*) – Unique search Id of the saved search.

- **page_number** (*int, optional*) – The consecutive group of results of size page_size in the result set to return. Defaults to None.
- **page_size** (*int, optional*) – The maximum number of results to be returned. Defaults to None.

Returns *py42.sdk.queries.fileevents.file_event_query.FileEventQuery*

Filter Classes

The following classes construct filters for file event queries. Each filter class corresponds to a file event detail. Call the appropriate classmethod on your desired filter class with the `value` you want to match and it will return a `FilterGroup` object that can be passed to `FileEventQuery`'s `all()` or `any()` methods to create complex queries that match multiple filter rules.

Example:

To search for events observed for certain set of documents, you can use the `FileName` and `MD5` filter classes to construct `FilterGroup`s that will search for matching filenames or (in case someone renamed the sensitive file) the known MD5 hashes of the files:

```
filename_filter = FileName.is_in(['confidential_plans.docx', 'confidential_plan_
↪projections.xlsx'])
md5_filter = MD5.is_in(['133765f4fff5e3038b9352a4d14e1532',
↪'ea16f0cbfc76f6eba292871f8a8c794b'])
```

See *Executing Searches* for more on building search queries.

Event Filters

`file_event_query.`**`create_exists_filter_group`**`()`

Creates a *FilterGroup* to find events where filter data exists. Useful for creating `EXISTS` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters **term** (*str*) – The term of the filter.

Returns *FilterGroup*

`file_event_query.`**`create_not_exists_filter_group`**`()`

Creates a *FilterGroup* to find events where filter data does not exist. Useful for creating `DOES_NOT_EXIST` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters **term** (*str*) – The term of the filter.

Returns *FilterGroup*

`file_event_query.`**`create_greater_than_filter_group`**`(`*value*`)`

Creates a *FilterGroup* for matching file events where the value with key `term` is greater than the given value. Useful for creating `GREATER_THAN` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters

- **term** (*str*) – The term of the filter.
- **value** (*str or int*) – The value used to filter file events.

Returns *FilterGroup*

file_event_query.**create_less_than_filter_group**(*value*)

Creates a *FilterGroup* for matching file events where the value with key `term` is less than the given value. Useful for creating `LESS_THAN` filters that are not yet supported in py42 or programmatically crafting filter groups.

Parameters

- **term** (*str*) – The term of the filter.
- **value** (*str or int*) – The value used to filter file events.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.event_filter.**EventTimestamp**

Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterTimestampField

Class that filters events based on the timestamp of the event that occurred.

Available event timestamp constants are provided as class attributes, These constants should be used only with class method *within_the_last*:

- `EventTimestamp.FIFTEEN_MINUTES`
- `EventTimestamp.ONE_HOUR`
- `EventTimestamp.THREE_HOURS`
- `EventTimestamp.TWELVE_HOURS`
- `EventTimestamp.ONE_DAY`
- `EventTimestamp.THREE_DAYS`
- `EventTimestamp.SEVEN_DAYS`
- `EventTimestamp.FOURTEEN_DAYS`
- `EventTimestamp.THIRTY_DAYS`

Example:: filter = EventTimestamp.within_the_last(EventTimestamp.SEVEN_DAYS)

classmethod in_range(*start_value*, *end_value*)

Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is in range between the provided `start_value` and `end_value`.

Parameters

- **start_value** (*str or int or float or datetime*) – The start value used to filter results.
- **end_value** (*str or int or float or datetime*) – The end value used to filter results.

Returns *FilterGroup*

classmethod on_or_after(*value*)

Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is on or after the provided ``value`.

Parameters **value** (*str or int or float or datetime*) – The value used to filter results.

Returns *FilterGroup*

classmethod on_or_before(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is on or before the provided `value`.

Parameters **value** (*str or int or float or datetime*) – The value used to filter results.

Returns *FilterGroup*

classmethod on_same_day(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is within the same calendar day as the provided `value`.

Parameters **value** (*str or int or float or datetime*) – The value used to filter results.

Returns *FilterGroup*

classmethod within_the_last(*value*)
Returns a *FilterGroup* that is useful for finding results where the key `self._term` is a timestamp-related term, such as `EventTimestamp._term`, and `value` is one of it's accepted values, such as one of the values in `EventTimestamp.choices()`.

Parameters **value** (*str*) – The value used to filter file events.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.event_filter.**EventType**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters file events based on event type.

Available event types are provided as class attributes:

- `EventType.CREATED`
- `EventType.DELETED`
- `EventType.EMAILED`
- `EventType.MODIFIED`
- `EventType.READ_BY_APP`
- `EventType.PRINTED`

Example:

```
filter = EventType.isin([EventType.READ_BY_APP, EventType.EMAILED])
```

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.event_filter.**InsertionTimestamp**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterTimestampField

Class that filters events based on the timestamp of when the event was actually added to the event store (which can be after the event occurred on the device itself).

value must be a POSIX timestamp. (see the *Dates* section of the Basics user guide for details on timestamp arguments in py42)

classmethod in_range(*start_value*, *end_value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is in range between the provided `start_value` and `end_value`.

Parameters

- **start_value** (*str or int or float or datetime*) – The start value used to filter results.
- **end_value** (*str or int or float or datetime*) – The end value used to filter results.

Returns *FilterGroup*

classmethod on_or_after(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is on or after the provided ``value`.

Parameters **value** (*str or int or float or datetime*) – The value used to filter results.

Returns *FilterGroup*

classmethod on_or_before(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is on or before the provided `value`.

Parameters **value** (*str or int or float or datetime*) – The value used to filter results.

Returns *FilterGroup*

classmethod on_same_day(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is within the same calendar day as the provided `value`.

Parameters **value** (*str or int or float or datetime*) – The value used to filter results.

Returns *FilterGroup*

classmethod within_the_last(*value*)
Returns a *FilterGroup* that is useful for finding results where the key `self._term` is a timestamp-related term, such as `EventTimestamp._term`, and `value` is one of it's accepted values, such as one of the values in `EventTimestamp.choices()`.

Parameters **value** (*str*) – The value used to filter file events.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.event_filter.**Source**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by event source.

Available source types are provided as class attributes:

- `Source.ENDPOINT`
- `Source.GOOGLE_DRIVE`
- `Source.ONE_DRIVE`
- `Source.BOX`
- `Source.GMAIL`
- `Source.OFFICE_365`

Example:

```
filter = Source.is_in([Source.ENDPOINT, Source.BOX])
```

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.event_filter.**MimeTypeMismatch**
Bases: *py42.sdk.queries.query_filter.QueryFilterBooleanField*

Class that filters events by whether or not a file's mime type matches its extension type.

classmethod is_false()
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is False.

Returns *FilterGroup*

classmethod is_true()
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is True.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.event_filter.**OutsideActiveHours**
Bases: *py42.sdk.queries.query_filter.QueryFilterBooleanField*

Class that filters events by whether or not they occurred outside a user's typical working hours

classmethod is_false()
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is False.

Returns *FilterGroup*

classmethod is_true()
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is True.

Returns *FilterGroup*

File Filters

class py42.sdk.queries.fileevents.filters.file_filter.**FileCategory**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by category of the file observed.

Available file categories are provided as class attributes:

- `FileCategory.AUDIO`
- `FileCategory.DOCUMENT`

- FileCategory.EXECUTABLE
- FileCategory.IMAGE
- FileCategory.PDF
- FileCategory.PRESENTATION
- FileCategory.SCRIPT
- FileCategory.SOURCE_CODE
- FileCategory.SPREADSHEET
- FileCategory.VIDEO
- FileCategory.VIRTUAL_DISK_IMAGE
- FileCategory.ZIP

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key self._term equals the provided value.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key self._term is in the provided value_list.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key self._term does not equal the provided value.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key self._term is not in the provided value_list.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.file_filter.**FileName**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by the name of the file observed.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.file_filter.**FileOwner**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by the owner of the file observed.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.file_filter.**FilePath**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by path of the file observed.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.file_filter.**FileSize**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterComparableField

Class that filters events by size of the file observed.

Size value must be bytes.

classmethod greater_than(*value*)
Returns a *FilterGroup* to find events where filter data is greater than the provided value.

Parameters **value** (*str or int or float*) – The value used to filter file events.

Returns *FilterGroup*

classmethod less_than(*value*)
Returns a *FilterGroup* to find events where filter data is less than than the provided value.

Parameters **value** (*str or int or float*) – The value used to filter file events.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.file_filter.**MD5**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by the MD5 hash of the file observed.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key self._term equals the provided value.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key self._term is in the provided value_list.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key self._term does not equal the provided value.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key self._term is not in the provided value_list.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.file_filter.**SHA256**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by SHA256 hash of the file observed.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

Device Filters

class py42.sdk.queries.fileevents.filters.device_filter.**DeviceUsername**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by the Code42 username of the device that observed the event.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.device_filter.**OSHostname**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by hostname of the device that observed the event.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.device_filter.**PrivateIPAddress**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by private (LAN) IP address of the device that observed the event.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.device_filter.**PublicIPAddress**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by public (WAN) IP address of the device that observed the event.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.device_filter.**DeviceSignedInUserName**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by signed in user of the device that observed the event.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

Cloud Filters

class py42.sdk.queries.fileevents.filters.cloud_filter.**Actor**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by the cloud service username of the event originator (applies to cloud data source events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.cloud_filter.**DirectoryID**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by unique identifier of the cloud drive or folder where the event occurred (applies to cloud data source events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.cloud_filter.**Shared**
Bases: *py42.sdk.queries.query_filter.QueryFilterBooleanField*

Class that filters events by the shared status of the file at the time the event occurred (applies to cloud data source events only).

classmethod is_false()
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is False.

Returns *FilterGroup*

classmethod is_true()
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is True.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.cloud_filter.**SharedWith**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by the list of users who had been granted access to the file at the time of the event (applies to cloud data source events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.cloud_filter.**SharingTypeAdded**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters results to include events where a file's sharing permissions were changed to a value that increases exposure (applies to cloud data source events only).

Available options provided as class attributes:

- `SharingTypeAdded.SHARED_VIA_LINK`
- `SharingTypeAdded.IS_PUBLIC`
- `SharingTypeAdded.OUTSIDE_TRUSTED_DOMAIN`

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

Exposure Filters

class py42.sdk.queries.fileevents.filters.exposure_filter.**ExposureType**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events based on exposure type.

Available options are provided as class attributes:

- `ExposureType.SHARED_VIA_LINK`
- `ExposureType.SHARED_TO_DOMAIN`
- `ExposureType.APPLICATION_READ`

- `ExposureType.CLOUD_STORAGE`
- `ExposureType.REMOVABLE_MEDIA`
- `ExposureType.IS_PUBLIC`

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.exposure_filter.**ProcessName**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events based on the process name involved in the exposure (applies to `read by browser or other app` events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.exposure_filter.**ProcessOwner**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events based on the process owner that was involved in the exposure (applies to `read by browser or other app` events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.exposure_filter.**RemovableMediaName**
Bases: `py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField`

Class that filters events based on the name of the removable media involved in the exposure (applies to `removable media` events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.exposure_filter.**RemovableMediaVendor**
Bases: `py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField`

Class that filters events based on the vendor of the removable media device involved in the exposure (applies to `removable media` events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.exposure_filter.**RemovableMediaMediaName**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events based on the name of the removable media (as reported by the vendor/device, usually very similar to RemovableMediaName) involved in the exposure (applies to `removable media` events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.exposure_filter.**RemovableMediaVolumeName**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events based on the name of the formatted volume (as reported by the operating system) of the removable media device involved in the exposure (applies to `removable media` events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.exposure_filter.**RemovableMediaPartitionID**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events based on the unique identifier assigned (by the operating system) to the removable media involved in the exposure (applies to `removable media` events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.exposure_filter.**RemovableMediaSerialNumber**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events based on the serial number of the connected hardware as reported by the operating system (applies to `removable media` events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.exposure_filter.**SyncDestination**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events based on the name of the cloud service the file is synced with (applies to `synced to cloud service` events only).

Available options are provided as class attributes:

- `SyncDestination.ICLOUD`
- `SyncDestination.BOX`
- `SyncDestination.BOX_DRIVE`
- `SyncDestination.GOOGLE_DRIVE`
- `SyncDestination.GOOGLE_BACKUP_AND_SYNC`
- `SyncDestination.DROPBOX`
- `SyncDestination.ONEDRIVE`

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.exposure_filter.**TabURL**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events based on the URL of the active browser tab at the time the file contents were read by the browser (applies to `read by browser or other app` events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.exposure_filter.**WindowTitle**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events based on the name of the browser tab or application window that was open when a browser or other app event occurred (applies to `read by browser or other app` events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

Email Filters

class py42.sdk.queries.fileevents.filters.email_filter.**EmailPolicyName**
Bases: *py42.sdk.queries.query_filter.QueryFilterStringField*

Class that filters events based on the email DLP policy that detected this file (applies to emails sent via Microsoft Office 365 only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.email_filter.**EmailSubject**
Bases: *py42.sdk.queries.query_filter.QueryFilterStringField*

Class that filters events based on the email's subject (applies to email events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.email_filter.**EmailRecipients**
Bases: *py42.sdk.queries.query_filter.QueryFilterStringField*

Class that filters events based on the email's recipient list (applies to email events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.email_filter.**EmailSender**
Bases: *py42.sdk.queries.query_filter.QueryFilterStringField*

Class that filters events based on the email's sender (applies to email events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.email_filter.**EmailFrom**
Bases: *py42.sdk.queries.query_filter.QueryFilterStringField*

Class that filters events based on the display name of the email's sender, as it appears in the "From:" field in the email (applies to email events only).

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

Activity Filters

class py42.sdk.queries.fileevents.filters.activity_filter.**TrustedActivity**
Bases: *py42.sdk.queries.query_filter.QueryFilterBooleanField*

Class that filters events based on whether activity can be trusted.

classmethod is_false()
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is False.

Returns *FilterGroup*

classmethod is_true()
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is True.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.activity_filter.**RemoteActivity**
Bases: *py42.sdk.queries.query_filter.QueryFilterBooleanField*

Class that filters events based on whether the activity was remote (took place outside of corporate IP range).

classmethod is_false()
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is False.

Returns *FilterGroup*

classmethod is_true()
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` is True.

Returns *FilterGroup*

Printer Filters

class py42.sdk.queries.fileevents.filters.print_filter.**Printer**
Bases: py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField

Class that filters events by printer name.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

class py42.sdk.queries.fileevents.filters.print_filter.**PrintJobName**
Bases: `py42.sdk.queries.fileevents.file_event_query.FileEventFilterStringField`

Class that filters events by print job name.

classmethod eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` equals the provided `value`.

Parameters **value** (*str*) – The value to match on.

Returns *FilterGroup*

classmethod exists()
Returns a *FilterGroup* to find events where filter data exists.

Returns *FilterGroup*

classmethod is_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to match on.

Returns *FilterGroup*

classmethod not_eq(*value*)
Returns a *FilterGroup* that is useful for finding results where the value with key `self._term` does not equal the provided `value`.

Parameters **value** (*str*) – The value to exclude on.

Returns *FilterGroup*

classmethod not_exists()
Returns a *FilterGroup* to find events where filter data does not exist.

Returns *FilterGroup*

classmethod not_in(*value_list*)
Returns a *FilterGroup* that is useful for finding results where the value with the key `self._term` is not in the provided `value_list`.

Parameters **value_list** (*list*) – The list of values to exclude on.

Returns *FilterGroup*

2.2.15 Archive

class py42.clients.archive.**ArchiveClient**(*archive_accessor_factory*, *archive_service*)
Bases: object

A module for getting information about backup archives on storage nodes along with functionality for streaming a file from backup.

get_all_by_device_guid(*device_guid*)
Gets archive information for a device. REST Documentation

Parameters **device_guid** (*str*) – The GUID for the device.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of archives.

Return type generator

get_all_device_restore_history(*days*, *device_id*)
Gets all restore jobs from the past given days for the device with the given ID. REST Documentation

Parameters

- **days** (*int*) – Number of days of restore history to retrieve.
- **device_id** (*int*) – The identification number of the device to get restore history for.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of restore history.

Return type generator

get_all_org_cold_storage_archives(*org_id*, *include_child_orgs=True*, *sort_key='archiveHoldExpireDate'*, *sort_dir='asc'*)
Returns a detailed list of cold storage archive information for a given org ID.

Parameters

- **org_id** (*str*) – The ID of a Code42 organization.
- **include_child_orgs** (*bool, optional*) – Determines whether cold storage information from the Org's children is also returned. Defaults to True.
- **sort_key** (*str, optional*) – Sets the property by which the returned results will be sorted. Choose from archiveHoldExpireDate, orgName, mountPointName, archiveBytes, and archiveType. Defaults to archiveHoldExpireDate.
- **sort_dir** (*str, optional*) – Sets the order by which sort_key should be sorted. Choose from asc or desc. Defaults to asc.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of cold storage archive information.

Return type generator

get_all_org_restore_history(*days*, *org_id*)
Gets all restore jobs from the past given days for the organization with the given ID. REST Documentation

Parameters

- **days** (*int*) – Number of days of restore history to retrieve.
- **org_id** (*int*) – The identification number of the organization to get restore history for.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of restore history.

Return type generator

get_all_user_restore_history(*days*, *user_id*)
Gets all restore jobs from the past given days for the user with the given ID. REST Documentation

Parameters

- **days** (*int*) – Number of days of restore history to retrieve.
- **user_id** (*int*) – The identification number of the user to get restore history for.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of restore history.

Return type generator

get_backup_sets(*device_guid*, *destination_guid*)
Gets all backup set names/identifiers referring to a single destination for a specific device. Learn more about backup sets.

Parameters

- **device_guid** (*str*) – The GUID of the device to get backup sets for.
- **destination_guid** (*str*) – The GUID of the destination containing the archive to get backup sets for.

Returns A response containing the backup sets.

Return type *py42.response.Py42Response*

get_by_archive_guid(*archive_guid*)
Gets single archive information by GUID. REST Documentation

Parameters **archive_guid** (*str*) – The GUID for the archive.

Returns A response containing archive information.

Return type *py42.response.Py42Response*

stream_from_backup(*file_paths*, *device_guid*, *destination_guid=None*, *archive_password=None*, *encryption_key=None*, *show_deleted=None*, *file_size_calc_timeout=10*)
Streams a file from a backup archive to memory. This method uses the same endpoint as restoring from Console and therefore has all the same considerations.

Support Documentation

Parameters

- **file_paths** (*str or list of str*) – The path or list of paths to the files or directories in the archive.
- **device_guid** (*str*) – The GUID of the device the file belongs to.
- **destination_guid** (*str, optional*) – The GUID of the destination that stores the backup of the file. If None, it will use the first destination GUID it finds for your device. 'destination_guid' may be useful if the file is missing from one of your destinations or if you want to optimize performance. Defaults to None.
- **archive_password** (*str or None, optional*) – The password for the archive, if password- protected. This is only relevant to users with archive key password security. Defaults to None.

- **encryption_key** (*str or None, optional*) – A custom encryption key for decrypting an archive's file contents, necessary for restoring files. This is only relevant to users with custom key archive security. Defaults to None.
- **show_deleted** (*bool, optional*) – Set to True to include deleted files when restoring a directory. Defaults to None.
- **file_size_calc_timeout** (*int, optional*) – Set to limit the amount of seconds spent calculating file sizes when crafting the request. Set to 0 or None to ignore file sizes altogether. Defaults to 10.

Returns A response containing the streamed content.

Return type *py42.response.Py42Response*

Usage example:

```
stream_response = sdk.archive.stream_from_backup("/full/path/to/file.txt",
↪"1234567890")
with open("/path/to/my/file", "wb") as f:
    for chunk in stream_response.iter_content(chunk_size=128):
        if chunk:
            f.write(chunk)
```

In certain cases, you will have to unzip the results:

```
import zipfile
with zipfile.ZipFile("downloaded_directory.zip", "r") as zf:
    zf.extractall(".")
```

stream_to_device(*file_paths*, *device_guid*, *accepting_device_guid*, *restore_path*, *destination_guid=None*, *archive_password=None*, *encryption_key=None*, *show_deleted=None*, *overwrite_existing_files=False*, *file_size_calc_timeout=10*)

Streams a file from a backup archive to a specified device.

Parameters

- **file_paths** (*str or list of str*) – The path or list of paths to the files or directories in the archive.
- **device_guid** (*str*) – The GUID of the device the file belongs to.
- **accepting_device_guid** (*str*) – The GUID of the device accepting the restore.
- **restore_path** (*str, optional*) – The path on the accepting device where the restore will be saved. Alternatively, pass in the value `ORIGINAL_LOCATION` to restore the file to the original location, which may be the case if you are replacing a device.
- **destination_guid** (*str, optional*) – The GUID of the destination that stores the backup of the file. If None, it will use the first destination GUID it finds for your device. 'destination_guid' may be useful if the file is missing from one of your destinations or if you want to optimize performance. Defaults to None.
- **archive_password** (*str or None, optional*) – The password for the archive, if password- protected. This is only relevant to users with archive key password security. Defaults to None.

- **encryption_key** (*str or None, optional*) – A custom encryption key for decrypting an archive's file contents, necessary for restoring files. This is only relevant to users with custom key archive security. Defaults to None.
- **show_deleted** (*bool, optional*) – Set to True to include deleted files when restoring a directory. Defaults to None.
- **overwrite_existing_files** (*bool, optional*) – to overwrite any existing files with the restored data. If False (the default), any existing files that match a path being restored will first get renamed.
- **file_size_calc_timeout** (*int, optional*) – Set to limit the amount of seconds spent calculating file sizes when crafting the request. Set to 0 or None to ignore file sizes altogether. Defaults to 10.

Returns *py42.response.Py42Response*.

update_cold_storage_purge_date(*archive_guid*, *purge_date*)
Updates the cold storage purge date for a specified archive. REST Documentation

Parameters

- **archive_guid** (*str*) – The identification number of the archive that should be updated
- **purge_date** (*str*) – The date on which the archive should be purged in yyyy-MM-dd format

Returns the response from the ColdStorage API.

Return type *py42.response.Py42Response*

2.2.16 Audit Logs

class py42.clients.auditlogs.**AuditLogsClient**(*audit_log_service*)
Bases: object

get_all(*begin_time=None*, *end_time=None*, *event_types=None*, *user_ids=None*, *usernames=None*, *user_ip_addresses=None*, *affected_user_ids=None*, *affected_usernames=None*, ***kwargs*)
Retrieve audit logs, filtered based on given arguments.

Parameters

- **begin_time** (*int or float or str or datetime, optional*) – Timestamp in milliseconds or str format "yyyy-MM-dd HH:MM:SS" or a datetime instance. Defaults to None.
- **end_time** (*int or float or str or datetime, optional*) – Timestamp in milliseconds or str format "yyyy-MM-dd HH:MM:SS" or a datetime instance. Defaults to None.
- **event_types** (*str or list, optional*) – A str or list of str of valid event types. Defaults to None.
- **user_ids** (*str or list, optional*) – A str or list of str of Code42 userUids. Defaults to None.
- **usernames** (*str or list, optional*) – A str or list of str of Code42 usernames. Defaults to None.
- **user_ip_addresses** (*str or list, optional*) – A str or list of str of user ip addresses. Defaults to None.

- **affected_user_ids** (*str or list, optional*) – A str or list of str of affected Code42 userUids. Defaults to None.
- **affected_usernames** (*str or list, optional*) – A str or list of str of affected Code42 usernames. Defaults to None.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of audit logs.

Return type generator

get_page(*page_num=1*, *page_size=None*, *begin_time=None*, *end_time=None*, *event_types=None*, *user_ids=None*, *usernames=None*, *user_ip_addresses=None*, *affected_user_ids=None*, *affected_usernames=None*, ***kwargs*)

Retrieve a page of audit logs, filtered based on given arguments.

Note: *page_num* here can be used same way as other methods that have a *page_num* parameter in py42. However, under the hood, it subtracts one from the given *page_num* in the implementation as the Code42 Audit-Logs API expects the start page to be zero.

Parameters

- **page_num** (*int, optional*) – The page number to get. Defaults to 1.
- **page_size** (*int, optional*) – The number of items per page. Defaults to *py42.settings.items_per_page*.
- **begin_time** (*int or float or str or datetime, optional*) – Timestamp in milliseconds or str format "yyyy-MM-dd HH:MM:SS" or a datetime instance. Defaults to None.
- **end_time** (*int or float or str or datetime, optional*) – Timestamp in milliseconds or str format "yyyy-MM-dd HH:MM:SS" or a datetime instance. Defaults to None.
- **event_types** (*str or list, optional*) – A str or list of str of valid event types. Defaults to None.
- **user_ids** (*str or list, optional*) – A str or list of str of Code42 userUids. Defaults to None.
- **usernames** (*str or list, optional*) – A str or list of str of Code42 usernames. Defaults to None.
- **user_ip_addresses** (*str or list, optional*) – A str or list of str of user ip addresses. Defaults to None.
- **affected_user_ids** (*str or list, optional*) – A str or list of str of affected Code42 userUids. Defaults to None.
- **affected_usernames** (*str or list, optional*) – A str or list of str of affected Code42 usernames. Defaults to None.

Returns *py42.response.Py42Response*

2.2.17 Cases

class py42.clients.cases.**CasesClient**(*cases_service*, *cases_file_event_service*)

Bases: object

A client to expose cases API.

Rest documentation

create(*name*, *subject=None*, *assignee=None*, *description=None*, *findings=None*)
Creates a new case.

Parameters

- **name** (*str*) – Name of the case.
- **subject** (*str, optional*) – User UID of a subject of a case.
- **assignee** (*str, optional*) – User UID of the assignee.
- **description** (*str, optional*) – Description of the case
- **findings** (*str, optional*) – Observations of the case.

Returns *py42.response.Py42Response*

export_summary(*case_number*)
Provides case summary to download as a PDF file.

Parameters **case_number** (*int*) – Case number of the case.

Returns *py42.response.Py42Response*

file_events
A collection of methods for managing file events associated with a given case.

Returns *py42.services.casesfileevents.CasesFileEventsService*

get(*case_number*)
Retrieve case details by case number.

Parameters **case_number** (*int*) – Case number of the case.

Returns *py42.response.Py42Response*

get_all(*name=None*, *status=None*, *min_create_time=None*, *max_create_time=None*, *min_update_time=None*, *max_update_time=None*, *subject=None*, *assignee=None*, *page_size=100*, *sort_direction='asc'*, *sort_key='number'*, ***kwargs*)
Gets all cases.

Parameters

- **name** (*str, optional*) – Filter results by case name, matches partial names. Defaults to None.
- **status** (*str, optional*) – Filter results by case status. *CaseStatus.OPEN* or *CaseStatus.CLOSED* Defaults to None.
- **min_create_time** (*str or int or float or datetime, optional*) – Filter results by case creation time, start time. str format %Y-%m-%d %H:%M:%S. Defaults to None.
- **max_create_time** (*str or int or float or datetime, optional*) – Filter results by case creation time, end time. str format %Y-%m-%d %H:%M:%S. Defaults to None.
- **min_update_time** (*str or int or float or datetime, optional*) – Filter results by last updated time, start time. str format %Y-%m-%d %H:%M:%S. Defaults to None.
- **max_update_time** (*str or int or float or datetime, optional*) – Filter results by last updated time, end time. str format %Y-%m-%d %H:%M:%S. Defaults to None.

- **subject** (*str, optional*) – Filter results based on User UID of a subject of a case. Defaults to None.
- **assignee** (*str, optional*) – Filter results based on User UID of an assignee of a case. Defaults to None.
- **page_size** (*int, optional*) – Number of results to return per page. Defaults to 100.
- **sort_direction** (*str, optional*) – The direction on which to sort the response, based on the corresponding sort key. *asc* or *desc*. Defaults to *asc*.
- **sort_key** (*str, optional*) – Values on which the response will be sorted. Defaults to "number". Available options are *name*, *number*, *createdAt*, *updatedAt*, *status*, *assigneeUsername*, *subjectUsername*.

Returns An object that iterates over *py42.response.Py42Response* objects that each contain a page of cases.

Return type generator

get_page(*page_num*, *name=None*, *status=None*, *min_create_time=None*, *max_create_time=None*, *min_update_time=None*, *max_update_time=None*, *subject=None*, *assignee=None*, *page_size=100*, *sort_direction='asc'*, *sort_key='number'*, ***kwargs*)

Gets individual page of cases.

Parameters

- **page_num** (*int*) – The page number to request.
- **name** (*str, optional*) – Filter results by case name, matches partial names. Defaults to None.
- **status** (*str, optional*) – Filter results by case status. *CaseStatus.OPEN* or *CaseStatus.CLOSED* Defaults to None.
- **min_create_time** (*str or int or float or datetime, optional*) – Filter results by case creation time, start time. str format %Y-%m-%d %H:%M:%S. Defaults to None.
- **max_create_time** (*str or int or float or datetime, optional*) – Filter results by case creation time, end time. str format %Y-%m-%d %H:%M:%S. Defaults to None.
- **min_update_time** (*str or int or float or datetime, optional*) – Filter results by last updated time, start time. str format %Y-%m-%d %H:%M:%S. Defaults to None.
- **max_update_time** (*str or int or float or datetime, optional*) – Filter results by last updated time, end time. str format %Y-%m-%d %H:%M:%S. Defaults to None.
- **subject** (*str, optional*) – Filter results based on User UID of a subject of a case. Defaults to None.
- **assignee** (*str, optional*) – Filter results based on User UID of an assignee of a case. Defaults to None.
- **page_size** (*int, optional*) – Number of results to return per page. Defaults to 100.
- **sort_direction** (*str, optional*) – The direction on which to sort the response, based on the corresponding sort key. *asc* or *desc*. Defaults to *asc*.

- **sort_key** (*str, optional*) – Values on which the response will be sorted. Defaults to "number". Available options are *name*, *number*, *createdAt*, *updatedAt*, *status*, *assigneeUsername*, *subjectUsername*.

Returns *py42.response.Py42Response*

update(*case_number*, *name=None*, *subject=None*, *assignee=None*, *description=None*, *findings=None*, *status=None*)
Updates case details for the given case number.

Parameters

- **case_number** (*int*) – Case number of the case.
- **name** (*str, optional*) – Name of the case. Defaults to None.
- **subject** (*str, optional*) – A subject of the case. Defaults to None.
- **assignee** (*str, optional*) – User UID of the assignee. Defaults to None.
- **description** (*str, optional*) – Description of the case. Defaults to None.
- **findings** (*str, optional*) – Notes on the case. Defaults to None.
- **status** (*str, optional*) – Status of the case. *CaseStatus.OPEN* or *CaseStatus.CLOSED*. Defaults to None.

Returns *py42.response.Py42Response*

Cases file events

class py42.services.casesfileevents.**CasesFileEventsService**(*connection*)
Bases: py42.services.BaseService

add(*case_number*, *event_id*)
Adds an event to the case.

Parameters

- **case_number** (*int*) – Case number of the case.
- **event_id** (*str*) – Event id to add to the case.

Returns *py42.response.Py42Response*

delete(*case_number*, *event_id*)
Deletes an event from the case.

Parameters

- **case_number** (*int*) – Case number of the case.
- **event_id** (*str*) – Event id to remove from case.

Returns *py42.response.Py42Response*

get(*case_number*, *event_id*)
Gets information of a specified event from the case.

Parameters

- **case_number** (*int*) – Case number of the case.
- **event_id** (*str*) – Event id to fetch from the case.

Returns *py42.response.Py42Response*

get_all(*case_number*)
Gets all events associated with the given case.

Parameters **case_number** (*int*) – Case number of the case.

Returns *py42.response.Py42Response*

2.2.18 Response

class py42.response.**Py42Response**(*requests_response*)
Bases: object

encoding
The encoding used to decode the response text.

headers
A case-insensitive dictionary of response headers.

iter_content(*chunk_size=1*, *decode_unicode=False*)
Iterates over the response data. When stream=True is set on the request, this avoids reading the content at once into memory for large responses.

Parameters

- **chunk_size** (*int, optional*) – The number of bytes it should read into memory. A value of None will function differently depending on the value of *stream*. stream=True will read data as it arrives in whatever size the chunks are received. If stream=False, data is returned as a single chunk. This is not necessarily the length of each item. Defaults to 1.
- **decode_unicode** (*bool, optional*) – If True, content will be decoded using the best available encoding based on the response. Defaults to False.

raw_text
The response.Response.text property. It contains raw metadata that is not included in the Py42Response.text property.

status_code
An integer code of the response HTTP Status, e.g. 404 or 200.

text
The more useful parts of the HTTP response dumped into a dictionary.

url
The final URL location of response.

2.2.19 Exceptions

exception py42.exceptions.**Py42ActiveLegalHoldError**(*exception*, *resource*, *resource_id*)
Bases: *py42.exceptions.Py42BadRequestError*

An exception raised when attempting to deactivate a user or device that is in an active legal hold.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42ArchiveFileNotFoundError**(*response*, *device_guid*, *file_path*)

Bases: *py42.exceptions.Py42ResponseError*

An exception raised when a resource file is not found or the path is invalid.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42BadRequestError**(*exception*, *message=None*)

Bases: *py42.exceptions.Py42HTTPError*

A wrapper to represent an HTTP 400 error.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42BadRestoreRequestError**(*exception*)

Bases: *py42.exceptions.Py42BadRequestError*

An error raised when the given restore arguments are not compatible and cause a bad request.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42CaseAlreadyHasEventError**(*exception*)

Bases: *py42.exceptions.Py42BadRequestError*

An error raised when event is already associated to the case.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42CaseNameExistsError**(*exception*, *case_name*)

Bases: *py42.exceptions.Py42BadRequestError*

An error raised when trying to create a case with a name that already exists.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42ChecksumNotFoundError**(*response*, *checksum_name*, *checksum_value*)

Bases: *py42.exceptions.Py42ResponseError*

An exception raised when a user-supplied hash could not successfully locate its corresponding resource.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42CloudAliasLimitExceededError**(*exception*, *message=None*)
Bases: *py42.exceptions.Py42BadRequestError*

An Exception raised when trying to add a cloud alias to a user when that user already has the max amount of supported cloud aliases.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42DescriptionLimitExceededError**(*exception*)
Bases: *py42.exceptions.Py42BadRequestError*

An error raised when description of a case exceeds the allowed char length limit.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42DeviceNotConnectedError**(*response*, *device_guid*)
Bases: *py42.exceptions.Py42ResponseError*

An exception raised when trying to push a restore to a device that is not connected to an Authority server.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42Error**
Bases: Exception

A generic, Py42 custom base exception.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42FeatureUnavailableError**(*response*)
Bases: *py42.exceptions.Py42ResponseError*

An exception raised when a requested feature is not supported in your Code42 environment.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42ForbiddenError**(*exception*, *message=None*)
Bases: *py42.exceptions.Py42HTTPError*

A wrapper to represent an HTTP 403 error.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42HTTPError**(*exception*, *message=None*)
Bases: *py42.exceptions.Py42ResponseError*

A base custom class to manage all HTTP errors raised by an API endpoint.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42InternalServerError**(*exception*, *message=None*)
Bases: *py42.exceptions.Py42HTTPError*

A wrapper to represent an HTTP 500 error.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42InvalidArchiveEncryptionKey**(*exception*)
Bases: *py42.exceptions.Py42HTTPError*

An exception raised the encryption key for an archive is invalid.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42InvalidArchivePassword**(*exception*)
Bases: *py42.exceptions.Py42HTTPError*

An exception raised when the password for unlocking an archive is invalid.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42InvalidPageTokenError**(*exception*, *page_token*)
Bases: *py42.exceptions.Py42BadRequestError*

An error raised when the page token given is invalid.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42InvalidRuleError**(*exception*, *rule_id*)
Bases: *py42.exceptions.Py42NotFoundError*

An exception raised when the observer rule ID does not exist.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42InvalidRuleOperationError**(*exception*, *rule_id*, *source*)
Bases: *py42.exceptions.Py42HTTPError*

An exception raised when trying to add or remove users to a system rule.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42LegalHoldCriteriaMissingError**(*exception*)
Bases: *py42.exceptions.Py42BadRequestError*

An exception raised when a bad request was made to a Legal Hold endpoint.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42LegalHoldNotFoundOrPermissionDeniedError**(*exception*, *matter_id*)
Bases: *py42.exceptions.Py42ForbiddenError*

An exception raised when a legal hold matter is inaccessible from your account or the matter ID is not valid.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42MFARequiredError**(*exception*, *message=None*)
Bases: *py42.exceptions.Py42UnauthorizedError*

An exception raised when a request requires multi-factor authentication

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42NotFoundError**(*exception*, *message=None*)
Bases: *py42.exceptions.Py42HTTPError*

A wrapper to represent an HTTP 404 error.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42ResponseError**(*response*, *message*)
Bases: *py42.exceptions.Py42Error*

A base custom class to manage all errors raised because of an HTTP response.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42SecurityPlanConnectionError**(*exception*, *error_message*)
Bases: *py42.exceptions.Py42HTTPError*

An exception raised when the user is not authorized to access the requested resource.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42SessionInitializationError**(*exception*)
Bases: *py42.exceptions.Py42Error*

An exception raised when a user connection is invalid. A connection might be invalid due to connection timeout, invalid token, etc.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42StorageSessionInitializationError**(*exception*, *error_message*)
Bases: *py42.exceptions.Py42HTTPError*

An exception raised when the user is not authorized to initialize a storage session. This may occur when trying to restore a file or trying to get events for file activity on removable media, in cloud sync folders, and browser uploads.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42TooManyRequestsError**(*exception*, *message=None*)
Bases: *py42.exceptions.Py42HTTPError*

A wrapper to represent an HTTP 429 error.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42UnauthorizedError**(*exception*, *message=None*)
Bases: *py42.exceptions.Py42HTTPError*

A wrapper to represent an HTTP 401 error.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42UpdateClosedCaseError**(*exception*)
Bases: *py42.exceptions.Py42BadRequestError*

An error raised when trying to update a closed case.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42UserAlreadyAddedError**(*exception*, *user_id*, *list_name*)
Bases: *py42.exceptions.Py42BadRequestError*

An exception raised when the user is already added to group or list, such as the Departing Employee list.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42UserAlreadyExistsError**(*exception*, *message=None*)
Bases: *py42.exceptions.Py42InternalServerError*

An exception raised when a user already exists

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception py42.exceptions.**Py42UserNotOnListError**(*exception*, *user_id*, *list_name*)
Bases: *py42.exceptions.Py42NotFoundError*

An exception raised when the user is not on a detection list.

response
The response prior to the error.

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

py42.exceptions.**raise_py42_error**(*raised_error*)
Raises the appropriate py42.exceptions.Py42HttpError based on the given HTTPError's response status code.

2.2.20 Util

py42.util.**convert_datetime_to_timestamp_str**(*date*)
Converts the given datetime to a formatted date str. The format matches strftime directives %Y-%m-%dT%H:%M:%S.%f.

Parameters **date** (*datetime*) – The datetime object to convert.

Returns A str representing the given date. Example output looks like '2020-03-25T15:29:04.465Z'.

Return type (str)

py42.util.**convert_timestamp_to_str**(*timestamp*)
Converts the given POSIX timestamp to a date str. The format matches strftime directives %Y-%m-%dT%H:%M:%S.%f.

Parameters **timestamp** (*float or int*) – A POSIX timestamp.

Returns A str representing the given timestamp. Example output looks like '2020-03-25T15:29:04.465Z'.

Return type (str)

py42.util.**format_json**(*json_string*)
Converts a minified JSON str to a prettified JSON str.

Parameters **json_string** (*str*) – A str representing minified JSON.

Returns A str representing prettified JSON.

Return type (str)

py42.util.**get_attribute_keys_from_class**(*cls*)
Returns attribute names for the given class.

Parameters **cls** (*class*) – The class to obtain attributes from.

Returns A list containing the attribute names of the given class.

Return type (list)

py42.util.**print_response**(*response*, *label=None*)
Prints a *py42.response.Py42Response* as prettified JSON. If unable to load, it prints the given response.

Parameters

- **response** (*py42.response.Py42Response*) – The response to print.
- **label** (*str, optional*) – A label at the beginning of the printed text. Defaults to None.

2.2.21 Shared Constants

class py42.constants.**SortDirection**
Bases: object

Code42 request *sort_direction* constants for sorting returned lists in responses.

py42.sdk.**from_local_account**(*host_address*, *username*, *password*, *totp=None*)
Creates a *SDKClient* object for accessing the Code42 REST APIs using the supplied credentials. Currently, only accounts created within the Code42 console or using the APIs (including py42) are supported. Username/passwords that are based on Active Directory, Okta, or other Identity providers cannot be used with this method.

Parameters

- **host_address** (*str*) – The domain name of the Code42 instance being authenticated to, e.g. console.us.code42.com
- **username** (*str*) – The username of the authenticating account.
- **password** (*str*) – The password of the authenticating account.
- **totp** (*callable or str, optional*) – The time-based one-time password of the authenticating account. Include only if the account uses Code42's two-factor authentication. Defaults to None.

Returns *py42.sdk.SDKClient*

class py42.sdk.**SDKClient**(*main_connection*, *auth*)
Bases: object

alerts
A collection of methods related to retrieving and updating alerts rules.

Returns py42.services.alertrules.AlertRulesClient

archive
A collection of methods for accessing Code42 storage archives. Useful for doing web-restores or finding a file on an archive.

Returns py42.services.archive.ArchiveClient

auditlogs
A collection of methods for retrieving audit logs.

Returns py42.services.auditlogs.AuditLogsService

cases
A collection of methods and properties for managing cases and file events associated with the case.

Returns py42.clients.cases.CaseClient

detectionlists
A collection of properties each containing methods for managing specific detection lists, such as departing employees.

Returns py42.services.detectionlists.DetectionListsClient

devices
A collection of methods for retrieving or updating data about devices in the Code42 environment.

Returns *py42.services.devices.DeviceService*

classmethod from_local_account(*host_address*, *username*, *password*, *totp=None*)
Creates a *SDKClient* object for accessing the Code42 REST APIs using the supplied credentials. Currently, only accounts created within the Code42 console or using the APIs (including py42) are supported. Username/passwords that are based on Active Directory, Okta, or other Identity providers cannot be used with this method.

Parameters

- **host_address** (*str*) – The domain name of the Code42 instance being authenticated to, e.g. console.us.code42.com
- **username** (*str*) – The username of the authenticating account.
- **password** (*str*) – The password of the authenticating account.
- **totp** (*callable or str, optional*) – The time-based one-time password of the authenticating account. Include only if the account uses Code42's two-factor authentication. Defaults to None.

Returns *py42.sdk.SDKClient*

legalhold
A collection of methods for retrieving and updating legal-hold matters, policies, and custodians.

Returns *py42.services.legalhold.LegalHoldService*

orgs
A collection of methods for retrieving or updating data about organizations in the Code42 environment.

Returns *py42.services.orgs.OrgService*

securitydata

- **File events**
  - Security plan information

Returns py42.services.securitydata.SecurityDataClient

Type A collection of methods and properties for getting security data such as

serveradmin
A collection of methods for getting server information for on-premise environments and tenant information for cloud environments.

Returns py42.services.administration.AdministrationService

usercontext
A collection of methods related to getting information about the currently logged in user, such as the tenant ID.

Returns *py42.usercontext.UserContext*

users
A collection of methods for retrieving or updating data about users in the Code42 environment.

Returns *py42.services.users.UserService*

Python Module Index

p

py42.exceptions, 99
py42.sdk, 106
py42.sdk.queries.alerts.filters.alert_filter, 42
py42.sdk.queries.query_filter, 53
py42.util, 105

Index

A

Actor (*class in py42.sdk.queries.alerts.filters.alert_filter*), 42
Actor (*class in py42.sdk.queries.fileevents.filters.cloud_filter*), 73
add() (*py42.services.casesfileevents.CasesFileEventsService method*), 98
add() (*py42.services.detectionlists.departing_employee.DepartingEmployeeService method*), 39
add() (*py42.services.detectionlists.high_risk_employee.HighRiskEmployeeService method*), 40
add_destination() (*py42.clients.settings.device_settings.BackupSet method*), 30
add_role() (*py42.services.users.UserService method*), 23
add_to_matter() (*py42.services.legalhold.LegalHoldService method*), 34
add_user() (*py42.clients.alertrules.AlertRulesClient method*), 51
add_user_cloud_alias() (*py42.clients.detectionlists.DetectionListsClient method*), 37
add_user_risk_tags() (*py42.clients.detectionlists.DetectionListsClient method*), 37
AlertQuery (*class in py42.sdk.queries.alerts.alert_query*), 49
AlertQueryFilterStringField (*class in py42.sdk.queries.alerts.filters.alert_filter*), 43
AlertRulesClient (*class in py42.clients.alertrules*), 51
alerts (*py42.sdk.SDKClient attribute*), 107
AlertsClient (*class in py42.clients.alerts*), 50
AlertState (*class in py42.sdk.queries.alerts.filters.alert_filter*), 43
archive (*py42.sdk.SDKClient attribute*), 107
archive_hold_days (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
ArchiveClient (*class in py42.clients.archive*), 91
auditlogs (*py42.sdk.SDKClient attribute*), 107
AuditLogsClient (*class in py42.clients.auditlogs*), 94
available_destinations (*py42.clients.settings.device_settings.DeviceSettingsDefaults attribute*), 23

B

backup_alert_recipient_emails (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
backup_critical_email_days (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
backup_sets (*py42.clients.settings.device_settings.DeviceSettings attribute*), 29
backup_status_email_enabled (*py42.clients.settings.device_settings.DeviceSettingsDefaults attribute*), 23
backup_status_email_frequency_days (*py42.clients.settings.device_settings.DeviceSettingsDefaults attribute*), 23
backup_warning_email_days (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
BackupSet (*class in py42.clients.settings.device_settings*), 30
block() (*py42.services.devices.DeviceService method*), 26
block() (*py42.services.orgs.OrgService method*), 18
block() (*py42.services.users.UserService method*), 23

C

cases (*py42.sdk.SDKClient attribute*), 107
CasesClient (*class in py42.clients.cases*), 95
CasesFileEventsService (*class in py42.services.casesfileevents*), 98

change_org_assignment() (*py42.services.users.UserService method*), 23
cloudshare (*py42.clients.alertrules.AlertRulesClient attribute*), 51
CloudShareService (*class in py42.services.alertrules*), 53
computer_id (*py42.clients.settings.device_settings.DeviceSettings attribute*), 29
contains() (*py42.sdk.queries.alerts.filters.alert_filter.Actor class method*), 42
contains() (*py42.sdk.queries.alerts.filters.alert_filter.AlertQueryFilterStringField class method*), 43
contains() (*py42.sdk.queries.alerts.filters.alert_filter.Description class method*), 45
contains() (*py42.sdk.queries.alerts.filters.alert_filter.RuleName class method*), 46
convert_datetime_to_timestamp_str() (*in module py42.util*), 105
convert_timestamp_to_str() (*in module py42.util*), 105
create() (*py42.clients.cases.CasesClient method*), 95
create_contains_filter_group() (*in module py42.sdk.queries.alerts.filters.alert_filter*), 49
create_eq_filter_group() (*in module py42.sdk.queries.query_filter*), 56
create_exists_filter_group() (*py42.sdk.queries.fileevents.file_event_query method*), 59
create_filter_group() (*in module py42.sdk.queries.query_filter*), 56
create_greater_than_filter_group() (*py42.sdk.queries.fileevents.file_event_query method*), 59
create_in_range_filter_group() (*in module py42.sdk.queries.query_filter*), 56
create_is_in_filter_group() (*in module py42.sdk.queries.query_filter*), 56
create_less_than_filter_group() (*py42.sdk.queries.fileevents.file_event_query method*), 59
create_matter() (*py42.services.legalhold.LegalHoldService method*), 34
create_not_contains_filter_group() (*in module py42.sdk.queries.alerts.filters.alert_filter*), 49
create_not_eq_filter_group() (*in module py42.sdk.queries.query_filter*), 56
create_not_exists_filter_group() (*py42.sdk.queries.fileevents.file_event_query method*), 59
create_not_in_filter_group() (*in module py42.sdk.queries.query_filter*), 57
create_on_or_after_filter_group() (*in module py42.sdk.queries.query_filter*), 57
create_on_or_before_filter_group() (*in module py42.sdk.queries.query_filter*), 57
create_org() (*py42.services.orgs.OrgService method*), 19
create_policy() (*py42.services.legalhold.LegalHoldService method*), 34
create_query_filter() (*in module py42.sdk.queries.query_filter*), 57
create_user() (*py42.clients.detectionlists.DetectionListsClient method*), 37
create_user() (*py42.services.users.UserService method*), 24
create_within_the_last_filter_group() (*in module py42.sdk.queries.query_filter*), 57
critical_alert_days (*py42.clients.settings.device_settings.DeviceSettingsDefaults attribute*), 23
critical_email_enabled (*py42.clients.settings.device_settings.DeviceSettingsDefaults attribute*), 23

D

DateObserved (*class in py42.sdk.queries.alerts.filters.alert_filter*), 44
deactivate() (*py42.services.devices.DeviceService method*), 26
deactivate() (*py42.services.orgs.OrgService method*), 19
deactivate() (*py42.services.users.UserService method*), 24
deactivate_matter() (*py42.services.legalhold.LegalHoldService method*), 34
deauthorize() (*py42.services.devices.DeviceService method*), 27
delete() (*py42.services.casesfileevents.CasesFileEventsService method*), 98
DepartingEmployeeFilters (*class in py42.services.detectionlists.departing_employee*), 39
DepartingEmployeeService (*class in py42.services.detectionlists.departing_employee*), 39
Description (*class in py42.sdk.queries.alerts.filters.alert_filter*), 45
destination_guid (*py42.clients.securitydata.PlanStorageInfo attribute*), 33
destinations (*py42.clients.settings.device_settings.BackupSet attribute*), 30
detectionlists (*py42.sdk.SDKClient attribute*), 107

DetectionListsClient (*class in py42.clients.detectionlists*), 37
device_id (*py42.clients.settings.device_settings.DeviceSettings attribute*), 29
devices (*py42.sdk.SDKClient attribute*), 107
DeviceService (*class in py42.services.devices*), 26
DeviceSettings (*class in py42.clients.settings.device_settings*), 29
DeviceSettingsDefaults (*class in py42.clients.settings.device_settings*), 22
DeviceSignedInUserName (*class in py42.sdk.queries.fileevents.filters.device_filter*), 72
DeviceUsername (*class in py42.sdk.queries.fileevents.filters.device_filter*), 69
DirectoryID (*class in py42.sdk.queries.fileevents.filters.cloud_filter*), 74

E

EmailFrom (*class in py42.sdk.queries.fileevents.filters.email_filter*), 88
EmailPolicyName (*class in py42.sdk.queries.fileevents.filters.email_filter*), 86
EmailRecipients (*class in py42.sdk.queries.fileevents.filters.email_filter*), 87
EmailSender (*class in py42.sdk.queries.fileevents.filters.email_filter*), 87
EmailSubject (*class in py42.sdk.queries.fileevents.filters.email_filter*), 86
encoding (*py42.response.Py42Response attribute*), 99
endpoint_monitoring_background_priority_enabled (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
endpoint_monitoring_browser_and_applications_enabled (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
endpoint_monitoring_cloud_sync_enabled (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
endpoint_monitoring_custom_applications_mac (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
endpoint_monitoring_custom_applications_win (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
endpoint_monitoring_enabled (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
endpoint_monitoring_file_exfiltration_detection_exc (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
endpoint_monitoring_file_metadata_collection_enable (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
endpoint_monitoring_file_metadata_collection_exclus (*py42.clients.settings.org_settings.OrgSettings attribute*), 21
endpoint_monitoring_file_metadata_ingest_scan_enabl (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
endpoint_monitoring_file_metadata_scan_enabled (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
endpoint_monitoring_printer_detection_enabled (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
endpoint_monitoring_removable_media_enabled (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
eq() (*py42.sdk.queries.alerts.filters.alert_filter.Actor class method*), 42
eq() (*py42.sdk.queries.alerts.filters.alert_filter.AlertQueryFilterStringFiel class method*), 43
eq() (*py42.sdk.queries.alerts.filters.alert_filter.AlertState class method*), 44
eq() (*py42.sdk.queries.alerts.filters.alert_filter.Description class method*), 45
eq() (*py42.sdk.queries.alerts.filters.alert_filter.RuleId class method*), 46
eq() (*py42.sdk.queries.alerts.filters.alert_filter.RuleName class method*), 46
eq() (*py42.sdk.queries.alerts.filters.alert_filter.RuleSource class method*), 47
eq() (*py42.sdk.queries.alerts.filters.alert_filter.RuleType class method*), 48
eq() (*py42.sdk.queries.alerts.filters.alert_filter.Severity class method*), 49
eq() (*py42.sdk.queries.fileevents.filters.cloud_filter.Actor class method*), 73
eq() (*py42.sdk.queries.fileevents.filters.cloud_filter.DirectoryID class method*), 74
eq() (*py42.sdk.queries.fileevents.filters.cloud_filter.SharedWith class method*), 75
eq() (*py42.sdk.queries.fileevents.filters.cloud_filter.SharingTypeAdded class method*), 76
eq() (*py42.sdk.queries.fileevents.filters.device_filter.DeviceSignedInUserN class method*), 72
eq() (*py42.sdk.queries.fileevents.filters.device_filter.DeviceUsername class method*), 69
eq() (*py42.sdk.queries.fileevents.filters.device_filter.OSHostname class method*), 70

eq() (*py42.sdk.queries.fileevents.filters.device_filter.PrivateIPAddress class method*), 71
eq() (*py42.sdk.queries.fileevents.filters.device_filter.PublicIPAddress class method*), 72
eq() (*py42.sdk.queries.fileevents.filters.email_filter.EmailFrom class method*), 88
eq() (*py42.sdk.queries.fileevents.filters.email_filter.EmailPolicyName class method*), 86
eq() (*py42.sdk.queries.fileevents.filters.email_filter.EmailRecipients class method*), 87
eq() (*py42.sdk.queries.fileevents.filters.email_filter.EmailSender class method*), 87
eq() (*py42.sdk.queries.fileevents.filters.email_filter.EmailSubject class method*), 86
eq() (*py42.sdk.queries.fileevents.filters.event_filter.EventType class method*), 61
eq() (*py42.sdk.queries.fileevents.filters.event_filter.Source class method*), 63
eq() (*py42.sdk.queries.fileevents.filters.exposure_filter.ExposureType class method*), 77
eq() (*py42.sdk.queries.fileevents.filters.exposure_filter.ProcessName class method*), 77
eq() (*py42.sdk.queries.fileevents.filters.exposure_filter.ProcessOwner class method*), 78
eq() (*py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaBusType class method*), 80
eq() (*py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaName class method*), 79
eq() (*py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaSerialNumber class method*), 82
eq() (*py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaVendor class method*), 82
eq() (*py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaVolumeName class method*), 79
eq() (*py42.sdk.queries.fileevents.filters.exposure_filter.SyncDestination class method*), 81
eq() (*py42.sdk.queries.fileevents.filters.exposure_filter.TabURL class method*), 83
eq() (*py42.sdk.queries.fileevents.filters.exposure_filter.WindowTitle class method*), 84
eq() (*py42.sdk.queries.fileevents.filters.file_filter.FileCategory class method*), 85
eq() (*py42.sdk.queries.fileevents.filters.file_filter.FileName class method*), 65
eq() (*py42.sdk.queries.fileevents.filters.file_filter.FileOwner class method*), 65
eq() (*py42.sdk.queries.fileevents.filters.file_filter.FilePath class method*), 66
eq() (*py42.sdk.queries.fileevents.filters.file_filter.MD5 class method*), 67
eq() (*py42.sdk.queries.fileevents.filters.file_filter.SHA256 class method*), 68
eq() (*py42.sdk.queries.fileevents.filters.print_filter.Printer class method*), 69
eq() (*py42.sdk.queries.fileevents.filters.print_filter.PrintJobName class method*), 89
eq() (*py42.sdk.queries.query_filter.QueryFilterStringField class method*), 90
eq() (*py42.sdk.queries.query_filter.QueryFilterStringField class method*), 54
EventTimestamp (*class in py42.sdk.queries.fileevents.filters.event_filter*), 60
EventType (*class in py42.sdk.queries.fileevents.filters.event_filter*), 61
excluded_files (*py42.clients.settings.device_settings.BackupSet attribute*), 30
execute() (*py42.services.savedsearch.SavedSearchService method*), 58
exfiltration (*py42.clients.alertrules.AlertRulesClient attribute*), 51
ExfiltrationService (*class in py42.services.alertrules*), 53
exists() (*py42.sdk.queries.fileevents.filters.cloud_filter.Actor class method*), 73
exists() (*py42.sdk.queries.fileevents.filters.cloud_filter.DirectoryID class method*), 74
exists() (*py42.sdk.queries.fileevents.filters.cloud_filter.SharedWith class method*), 75
exists() (*py42.sdk.queries.fileevents.filters.cloud_filter.SharingTypeAdded class method*), 76
exists() (*py42.sdk.queries.fileevents.filters.device_filter.DeviceSignedIn class method*), 72
exists() (*py42.sdk.queries.fileevents.filters.device_filter.DeviceUsername class method*), 70
exists() (*py42.sdk.queries.fileevents.filters.device_filter.OSHostname class method*), 70
exists() (*py42.sdk.queries.fileevents.filters.device_filter.PrivateIPAddress class method*), 71
exists() (*py42.sdk.queries.fileevents.filters.device_filter.PublicIPAddress class method*), 72
exists() (*py42.sdk.queries.fileevents.filters.event_filter.EventType class method*), 61
exists() (*py42.sdk.queries.fileevents.filters.event_filter.Source class method*), 63
exists() (*py42.sdk.queries.fileevents.filters.exposure_filter.ExposureType class method*), 77
exists() (*py42.sdk.queries.fileevents.filters.exposure_filter.ProcessName class method*), 77
exists() (*py42.sdk.queries.fileevents.filters.exposure_filter.ProcessOwner class method*), 78
exists() (*py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaBusType class method*), 80
exists() (*py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaName class method*), 79
exists() (*py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaSerialNumber class method*), 82

exists() (*py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMedia class method*), 83
exists() (*py42.sdk.queries.fileevents.filters.exposure_filter.ReportedCloudSharing class method*), 80
exists() (*py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaVendorProduct class method*), 81
exists() (*py42.sdk.queries.fileevents.filters.exposure_filter.SyncDestination class method*), 83
exists() (*py42.sdk.queries.fileevents.filters.exposure_filter.TabURL class method*), 84
exists() (*py42.sdk.queries.fileevents.filters.exposure_filter.WindowTitle class method*), 85
exists() (*py42.sdk.queries.fileevents.filters.file_filter.FileCategory class method*), 65
exists() (*py42.sdk.queries.fileevents.filters.file_filter.FileName class method*), 66
exists() (*py42.sdk.queries.fileevents.filters.file_filter.FileOwner class method*), 66
exists() (*py42.sdk.queries.fileevents.filters.file_filter.FilePath class method*), 67
exists() (*py42.sdk.queries.fileevents.filters.file_filter.MD5 class method*), 68
exists() (*py42.sdk.queries.fileevents.filters.file_filter.SHA256 class method*), 69
exists() (*py42.sdk.queries.fileevents.filters.print_filter.Printer class method*), 89
exists() (*py42.sdk.queries.fileevents.filters.print_filter.PrintJobName class method*), 90
export_summary() (*py42.clients.cases.CasesClient method*), 96
ExposureType (*class in py42.sdk.queries.fileevents.filters.exposure_filter*), 76
external_reference (*py42.clients.settings.device_settings.DeviceSettings attribute*), 29
external_reference (*py42.clients.settings.org_settings.OrgSettings attribute*), 22

F

file_events (*py42.clients.cases.CasesClient attribute*), 96
FileCategory (*class in py42.sdk.queries.fileevents.filters.file_filter*), 64
FileEventQuery (*class in py42.sdk.queries.fileevents.file_event_query*), 58
FileName (*class in py42.sdk.queries.fileevents.filters.file_filter*), 65
filename_exclusions (*py42.clients.settings.device_settings.BackupSet attribute*), 30
FileOwner (*class in py42.sdk.queries.fileevents.filters.file_filter*), 66
FilePath (*class in py42.sdk.queries.fileevents.filters.file_filter*), 67
FileTypeMismatchService (*class in py42.services.alertrules*), 53
filter_clause (*py42.sdk.queries.query_filter.FilterGroup attribute*), 53
filter_list (*py42.sdk.queries.query_filter.FilterGroup attribute*), 53
FilterGroup (*class in py42.sdk.queries.query_filter*), 53
format_json() (*in module py42.util*), 106
from_dict() (*py42.sdk.queries.query_filter.FilterGroup class method*), 53
from_dict() (*py42.sdk.queries.query_filter.QueryFilter class method*), 54
from_local_account() (*in module py42.sdk*), 106
from_local_account() (*py42.sdk.SDKClient class method*), 107

G

get() (*py42.clients.cases.CasesClient method*), 96
get() (*py42.services.alertrules.CloudShareService method*), 53
get() (*py42.services.alertrules.ExfiltrationService method*), 53
get() (*py42.services.alertrules.FileTypeMismatchService method*), 53
get() (*py42.services.casesfileevents.CasesFileEventsService method*), 98
get() (*py42.services.detectionlists.departing_employee.DepartingEmploy method*), 39
get() (*py42.services.detectionlists.high_risk_employee.HighRiskEmploye method*), 41
get() (*py42.services.savedsearch.SavedSearchService method*), 58
get_agent_full_disk_access_state() (*py42.services.devices.DeviceService method*), 27
get_agent_full_disk_access_states() (*py42.services.orgs.OrgService method*), 19
get_agent_state() (*py42.services.devices.DeviceService method*), 27
get_agent_state() (*py42.services.orgs.OrgService method*), 19
get_all() (*py42.clients.alertrules.AlertRulesClient method*), 51

get_all() (*py42.clients.auditlogs.AuditLogsClient method*), 94
get_all() (*py42.clients.cases.CasesClient method*), 96
get_all() (*py42.services.casesfileevents.CasesFileEventsService method*), 98
get_all() (*py42.services.detectionlists.departing_employee.DepartingEmployeeService method*), 39
get_all() (*py42.services.detectionlists.high_risk_employee.HighRiskEmployeeService method*), 41
get_all() (*py42.services.devices.DeviceService method*), 27
get_all() (*py42.services.orgs.OrgService method*), 19
get_all() (*py42.services.users.UserService method*), 24
get_all_by_device_guid() (*py42.clients.archive.ArchiveClient method*), 91
get_all_by_name() (*py42.clients.alertrules.AlertRulesClient method*), 52
get_all_device_restore_history() (*py42.clients.archive.ArchiveClient method*), 91
get_all_matter_custodians() (*py42.services.legalhold.LegalHoldService method*), 35
get_all_matters() (*py42.services.legalhold.LegalHoldService method*), 35
get_all_org_cold_storage_archives() (*py42.clients.archive.ArchiveClient method*), 91
get_all_org_restore_history() (*py42.clients.archive.ArchiveClient method*), 91
get_all_plan_security_events() (*py42.clients.securitydata.SecurityDataClient method*), 31
get_all_user_restore_history() (*py42.clients.archive.ArchiveClient method*), 92
get_all_user_security_events() (*py42.clients.securitydata.SecurityDataClient method*), 32
get_attribute_keys_from_class() (*in module py42.util*), 106
get_available_roles() (*py42.services.users.UserService method*), 24
get_backup_sets() (*py42.clients.archive.ArchiveClient method*), 92
get_by_archive_guid() (*py42.clients.archive.ArchiveClient method*), 92
get_by_guid() (*py42.services.devices.DeviceService method*), 28
get_by_id() (*py42.services.devices.DeviceService method*), 28
get_by_id() (*py42.services.orgs.OrgService method*), 19
get_by_id() (*py42.services.savedsearch.SavedSearchService method*), 58
get_by_id() (*py42.services.users.UserService method*), 25
get_by_observer_id() (*py42.clients.alertrules.AlertRulesClient method*), 52
get_by_uid() (*py42.services.orgs.OrgService method*), 20
get_by_uid() (*py42.services.users.UserService method*), 25
get_by_username() (*py42.services.users.UserService method*), 25
get_current() (*py42.services.orgs.OrgService method*), 20
get_current() (*py42.services.users.UserService method*), 25
get_current_tenant_id() (*py42.usercontext.UserContext method*), 26
get_custodians_page() (*py42.services.legalhold.LegalHoldService method*), 35
get_details() (*py42.clients.alerts.AlertsClient method*), 50
get_matter_by_uid() (*py42.services.legalhold.LegalHoldService method*), 36
get_matters_page() (*py42.services.legalhold.LegalHoldService method*), 36
get_page() (*py42.clients.alertrules.AlertRulesClient method*), 52
get_page() (*py42.clients.auditlogs.AuditLogsClient method*), 95
get_page() (*py42.clients.cases.CasesClient method*), 97
get_page() (*py42.services.detectionlists.departing_employee.Departing method*), 39
get_page() (*py42.services.detectionlists.high_risk_employee.HighRiskE method*), 41
get_page() (*py42.services.devices.DeviceService method*), 28
get_page() (*py42.services.orgs.OrgService method*),

20
get_page() (py42.services.users.UserService method), 25
get_policy_by_uid() (py42.services.legalhold.LegalHoldService method), 37
get_policy_list() (py42.services.legalhold.LegalHoldService method), 37
get_query() (py42.services.savedsearch.SavedSearchService method), 58
get_roles() (py42.services.users.UserService method), 25
get_scim_data_by_uid() (py42.services.users.UserService method), 26
get_security_plan_storage_info_list() (py42.clients.securitydata.SecurityDataClient method), 32
get_settings() (py42.services.devices.DeviceService method), 29
get_settings() (py42.services.orgs.OrgService method), 20
get_user() (py42.clients.detectionlists.DetectionListsClient method), 38
get_user_by_id() (py42.clients.detectionlists.DetectionListsClient method), 38
greater_than() (py42.sdk.queries.fileevents.filters.file_filter.FileSize class method), 68
guid (py42.clients.settings.device_settings.DeviceSettings attribute), 29

H

headers (py42.response.Py42Response attribute), 99
HighRiskEmployeeFilters (class in py42.services.detectionlists.high_risk_employee), 40
HighRiskEmployeeService (class in py42.services.detectionlists.high_risk_employee), 40

I

in_range() (py42.sdk.queries.alerts.filters.alert_filter.DateObserved class method), 44
in_range() (py42.sdk.queries.fileevents.filters.event_filter.EventTimestamp class method), 60
in_range() (py42.sdk.queries.fileevents.filters.event_filter.InsertionTimestamp class method), 62
in_range() (py42.sdk.queries.query_filter.QueryFilterTimestampField class method), 55
included_files (py42.clients.settings.device_settings.BackupSet attribute), 30
InsertionTimestamp (class in py42.sdk.queries.fileevents.filters.event_filter), 62
is_false() (py42.sdk.queries.fileevents.filters.activity_filter.RemoteActi class method), 89
is_false() (py42.sdk.queries.fileevents.filters.activity_filter.TrustedActi class method), 88
is_false() (py42.sdk.queries.fileevents.filters.cloud_filter.Shared class method), 74
is_false() (py42.sdk.queries.fileevents.filters.event_filter.MimeTypeMis class method), 64
is_false() (py42.sdk.queries.fileevents.filters.event_filter.OutsideActive class method), 64
is_false() (py42.sdk.queries.query_filter.QueryFilterBooleanField class method), 54
is_in() (py42.sdk.queries.alerts.filters.alert_filter.Actor class method), 42
is_in() (py42.sdk.queries.alerts.filters.alert_filter.AlertQueryFilterString class method), 43
is_in() (py42.sdk.queries.alerts.filters.alert_filter.AlertState class method), 44
is_in() (py42.sdk.queries.alerts.filters.alert_filter.Description class method), 45
is_in() (py42.sdk.queries.alerts.filters.alert_filter.RuleId class method), 46
is_in() (py42.sdk.queries.alerts.filters.alert_filter.RuleName class method), 47
is_in() (py42.sdk.queries.alerts.filters.alert_filter.RuleSource class method), 47
is_in() (py42.sdk.queries.alerts.filters.alert_filter.RuleType class method), 48
is_in() (py42.sdk.queries.alerts.filters.alert_filter.Severity class method), 49
is_in() (py42.sdk.queries.fileevents.filters.cloud_filter.Actor class method), 73
is_in() (py42.sdk.queries.fileevents.filters.cloud_filter.DirectoryID class method), 74
is_in() (py42.sdk.queries.fileevents.filters.cloud_filter.SharedWith class method), 75
is_in() (py42.sdk.queries.fileevents.filters.cloud_filter.SharingTypeAdde class method), 76
is_in() (py42.sdk.queries.fileevents.filters.device_filter.DeviceSignedInU class method), 72
is_in() (py42.sdk.queries.fileevents.filters.device_filter.DeviceUsername class method), 70
is_in() (py42.sdk.queries.fileevents.filters.device_filter.OSHostname class method), 70
is_in() (py42.sdk.queries.fileevents.filters.device_filter.PrivateIPAddress class method), 71
is_in() (py42.sdk.queries.fileevents.filters.device_filter.PublicIPAddress class method), 72
is_in() (py42.sdk.queries.fileevents.filters.email_filter.EmailFrom class method), 88
is_in() (py42.sdk.queries.fileevents.filters.email_filter.EmailPolicyName class method), 86
is_in() (py42.sdk.queries.fileevents.filters.email_filter.EmailRecipients

class method), 87
is_in() (py42.sdk.queries.fileevents.filters.email_filter.EmailSender class method), 87
is_in() (py42.sdk.queries.fileevents.filters.email_filter.EmailSubject class method), 86
is_in() (py42.sdk.queries.fileevents.filters.event_filter.EventType class method), 61
is_in() (py42.sdk.queries.fileevents.filters.event_filter.Source class method), 63
is_in() (py42.sdk.queries.fileevents.filters.exposure_filter.ExposureType class method), 77
is_in() (py42.sdk.queries.fileevents.filters.exposure_filter.ProcessName class method), 77
is_in() (py42.sdk.queries.fileevents.filters.exposure_filter.ProcessOwner class method), 78
is_in() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaName class method), 80
is_in() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaPartitionID class method), 79
is_in() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaSerialNumber class method), 82
is_in() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaVendor class method), 83
is_in() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaVolumeName class method), 80
is_in() (py42.sdk.queries.fileevents.filters.exposure_filter.SyncDestination class method), 81
is_in() (py42.sdk.queries.fileevents.filters.exposure_filter.TabURL class method), 84
is_in() (py42.sdk.queries.fileevents.filters.exposure_filter.WindowTitle class method), 84
is_in() (py42.sdk.queries.fileevents.filters.file_filter.FileCategory class method), 85
is_in() (py42.sdk.queries.fileevents.filters.file_filter.FileName class method), 65
is_in() (py42.sdk.queries.fileevents.filters.file_filter.FileOwner class method), 66
is_in() (py42.sdk.queries.fileevents.filters.file_filter.FilePath class method), 66
is_in() (py42.sdk.queries.fileevents.filters.file_filter.MD5 class method), 67
is_in() (py42.sdk.queries.fileevents.filters.file_filter.SHA256 class method), 68
is_in() (py42.sdk.queries.fileevents.filters.print_filter.Printer class method), 69
is_in() (py42.sdk.queries.fileevents.filters.print_filter.PrintJobName class method), 89
is_in() (py42.sdk.queries.query_filter.QueryFilterStringField class method), 90
is_true() (py42.sdk.queries.fileevents.filters.activity_filter.RemoteActivity class method), 55
is_true() (py42.sdk.queries.fileevents.filters.activity_filter.TrustedActivity class method), 89
is_true() (py42.sdk.queries.fileevents.filters.cloud_filter.SharedToDomains class method), 89
is_true() (py42.sdk.queries.fileevents.filters.event_filter.MimeTypeMismatch class method), 75
is_true() (py42.sdk.queries.fileevents.filters.event_filter.OutsideActiveHours class method), 64
is_true() (py42.sdk.queries.query_filter.QueryFilterBooleanField class method), 64
is_true() (py42.sdk.queries.query_filter.QueryFilterBooleanField class method), 54
issue_current() (py42.response.Py42Response method), 99

J

java_memory_heap_max (py42.clients.settings.device_settings.DeviceSettings attribute), 30

L

legalhold (py42.sdk.SDKClient attribute), 107
LegalHoldService (class in py42.services.legalhold), 34
less_than() (py42.sdk.queries.fileevents.filters.file_filter.FileSize class method), 68
lock_destination() (py42.clients.settings.device_settings.BackupSet method), 30
locked (py42.clients.settings.device_settings.BackupSet attribute), 30

M

maximum_user_subscriptions (py42.clients.settings.org_settings.OrgSettings attribute), 22
MD5 (class in py42.sdk.queries.fileevents.filters.file_filter), 68
MimeTypeMismatch (class in py42.sdk.queries.fileevents.filters.event_filter), 64

N

name (py42.clients.settings.device_settings.DeviceSettings attribute), 30
node_guid (py42.clients.securitydata.PlanStorageInfo attribute), 33
not_contains() (py42.sdk.queries.alerts.filters.alert_filter.Actor class method), 42
not_contains() (py42.sdk.queries.alerts.filters.alert_filter.AlertQuery class method), 43
not_contains() (py42.sdk.queries.alerts.filters.alert_filter.Description class method), 45
not_contains() (py42.sdk.queries.alerts.filters.alert_filter.RuleName class method), 47
not_eq() (py42.sdk.queries.alerts.filters.alert_filter.Actor class method), 42

not_eq() (py42.sdk.queries.alerts.filters.alert_filter.AlertQueryFilterStringField class method), 43
not_eq() (py42.sdk.queries.alerts.filters.alert_filter.AlertState class method), 44
not_eq() (py42.sdk.queries.alerts.filters.alert_filter.Description class method), 45
not_eq() (py42.sdk.queries.alerts.filters.alert_filter.RuleId class method), 46
not_eq() (py42.sdk.queries.alerts.filters.alert_filter.RuleName class method), 47
not_eq() (py42.sdk.queries.alerts.filters.alert_filter.RuleSource class method), 47
not_eq() (py42.sdk.queries.alerts.filters.alert_filter.RuleType class method), 48
not_eq() (py42.sdk.queries.alerts.filters.alert_filter.Severity class method), 49
not_eq() (py42.sdk.queries.fileevents.filters.cloud_filter.Actor class method), 73
not_eq() (py42.sdk.queries.fileevents.filters.cloud_filter.DirectoryID class method), 74
not_eq() (py42.sdk.queries.fileevents.filters.cloud_filter.SharedWith class method), 75
not_eq() (py42.sdk.queries.fileevents.filters.cloud_filter.SharingTypeAdded class method), 76
not_eq() (py42.sdk.queries.fileevents.filters.device_filter.DeviceSignedByName class method), 73
not_eq() (py42.sdk.queries.fileevents.filters.device_filter.DeviceUsername class method), 70
not_eq() (py42.sdk.queries.fileevents.filters.device_filter.OSHostname class method), 70
not_eq() (py42.sdk.queries.fileevents.filters.device_filter.PrivateIPAddress class method), 71
not_eq() (py42.sdk.queries.fileevents.filters.device_filter.PublicIPAddress class method), 72
not_eq() (py42.sdk.queries.fileevents.filters.email_filter.EmailFrom class method), 88
not_eq() (py42.sdk.queries.fileevents.filters.email_filter.EmailPolicyName class method), 86
not_eq() (py42.sdk.queries.fileevents.filters.email_filter.EmailRecipients class method), 87
not_eq() (py42.sdk.queries.fileevents.filters.email_filter.EmailSender class method), 88
not_eq() (py42.sdk.queries.fileevents.filters.email_filter.EmailSubject class method), 86
not_eq() (py42.sdk.queries.fileevents.filters.event_filter.EventType class method), 62
not_eq() (py42.sdk.queries.fileevents.filters.event_filter.Source class method), 63
not_eq() (py42.sdk.queries.fileevents.filters.exposure_filter.ExposureType class method), 77
not_eq() (py42.sdk.queries.fileevents.filters.exposure_filter.ProcessName class method), 78
not_eq() (py42.sdk.queries.fileevents.filters.exposure_filter.ProcessOwner class method), 78
not_eq() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableM class method), 81
not_eq() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableM class method), 79
not_eq() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableM class method), 82
not_eq() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableM class method), 83
not_eq() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableM class method), 80
not_eq() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableM class method), 81
not_eq() (py42.sdk.queries.fileevents.filters.exposure_filter.SyncDestinat class method), 84
not_eq() (py42.sdk.queries.fileevents.filters.exposure_filter.TabURL class method), 84
not_eq() (py42.sdk.queries.fileevents.filters.exposure_filter.WindowTitle class method), 85
not_eq() (py42.sdk.queries.fileevents.filters.file_filter.FileCategory class method), 65
not_eq() (py42.sdk.queries.fileevents.filters.file_filter.FileName class method), 66
not_eq() (py42.sdk.queries.fileevents.filters.file_filter.FileOwner class method), 67
not_eq() (py42.sdk.queries.fileevents.filters.file_filter.FilePath class method), 67
not_eq() (py42.sdk.queries.fileevents.filters.file_filter.MD5 class method), 68
not_eq() (py42.sdk.queries.fileevents.filters.file_filter.SHA256 class method), 69
not_eq() (py42.sdk.queries.fileevents.filters.print_filter.Printer class method), 89
not_eq() (py42.sdk.queries.fileevents.filters.print_filter.PrintJobName class method), 90
not_eq() (py42.sdk.queries.query_filter.QueryFilterStringField class method), 55
not_eq() (py42.sdk.queries.fileevents.filters.cloud_filter.Actor class method), 73
not_eq() (py42.sdk.queries.fileevents.filters.cloud_filter.DirectoryI class method), 74
not_exists() (py42.sdk.queries.fileevents.filters.cloud_filter.SharedWit class method), 75
not_exists() (py42.sdk.queries.fileevents.filters.cloud_filter.SharingTyp class method), 76
not_exists() (py42.sdk.queries.fileevents.filters.device_filter.DeviceSig class method), 73
not_exists() (py42.sdk.queries.fileevents.filters.device_filter.DeviceUs class method), 70
not_exists() (py42.sdk.queries.fileevents.filters.device_filter.OSHostna class method), 71
not_exists() (py42.sdk.queries.fileevents.filters.device_filter.PrivateIP class method), 71
not_exists() (py42.sdk.queries.fileevents.filters.device_filter.PublicIPA class method), 72

not_exists() (py42.sdk.queries.fileevents.filters.event_filter.EventType class method), 62
not_exists() (py42.sdk.queries.fileevents.filters.event_filter.Source class method), 64
not_exists() (py42.sdk.queries.fileevents.filters.exposure_filter.ExposureType class method), 77
not_exists() (py42.sdk.queries.fileevents.filters.exposure_filter.ProcessName class method), 78
not_exists() (py42.sdk.queries.fileevents.filters.exposure_filter.ProcessOwner class method), 78
not_exists() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaMediaName class method), 81
not_exists() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaName class method), 79
not_exists() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaPartitionID class method), 82
not_exists() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaSerialNumber class method), 83
not_exists() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaVendor class method), 80
not_exists() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaVolumeName class method), 81
not_exists() (py42.sdk.queries.fileevents.filters.exposure_filter.SyncDestination class method), 84
not_exists() (py42.sdk.queries.fileevents.filters.exposure_filter.TabURL class method), 85
not_exists() (py42.sdk.queries.fileevents.filters.exposure_filter.WindowTitle class method), 85
not_exists() (py42.sdk.queries.fileevents.filters.file_filter.FileCategory class method), 65
not_exists() (py42.sdk.queries.fileevents.filters.file_filter.FileName class method), 66
not_exists() (py42.sdk.queries.fileevents.filters.file_filter.FileOwner class method), 67
not_exists() (py42.sdk.queries.fileevents.filters.file_filter.FilePath class method), 67
not_exists() (py42.sdk.queries.fileevents.filters.file_filter.MD5 class method), 68
not_exists() (py42.sdk.queries.fileevents.filters.file_filter.SHA256 class method), 69
not_exists() (py42.sdk.queries.fileevents.filters.print_filter.Printer class method), 90
not_exists() (py42.sdk.queries.fileevents.filters.print_filter.PrintJobName class method), 90
not_in() (py42.sdk.queries.alerts.filters.alert_filter.Actor class method), 42
not_in() (py42.sdk.queries.alerts.filters.alert_filter.AlertQueryFilterStringField class method), 43
not_in() (py42.sdk.queries.alerts.filters.alert_filter.AlertState class method), 44
not_in() (py42.sdk.queries.alerts.filters.alert_filter.Description class method), 46
not_in() (py42.sdk.queries.alerts.filters.alert_filter.RuleId class method), 46
not_in() (py42.sdk.queries.alerts.filters.alert_filter.RuleName class method), 47
not_in() (py42.sdk.queries.alerts.filters.alert_filter.RuleSource class method), 48
not_in() (py42.sdk.queries.alerts.filters.alert_filter.RuleType class method), 48
not_in() (py42.sdk.queries.alerts.filters.alert_filter.Severity class method), 49
not_in() (py42.sdk.queries.fileevents.filters.cloud_filter.Actor class method), 74
not_in() (py42.sdk.queries.fileevents.filters.cloud_filter.DirectoryID class method), 74
not_in() (py42.sdk.queries.fileevents.filters.cloud_filter.SharedWith class method), 75
not_in() (py42.sdk.queries.fileevents.filters.cloud_filter.SharingTypeAdded class method), 76
not_in() (py42.sdk.queries.fileevents.filters.device_filter.DeviceSignedIn class method), 73
not_in() (py42.sdk.queries.fileevents.filters.device_filter.DeviceUsername class method), 70
not_in() (py42.sdk.queries.fileevents.filters.device_filter.OSHostname class method), 71
not_in() (py42.sdk.queries.fileevents.filters.device_filter.PrivateIPAddress class method), 71
not_in() (py42.sdk.queries.fileevents.filters.device_filter.PublicIPAddress class method), 72
not_in() (py42.sdk.queries.fileevents.filters.email_filter.EmailFrom class method), 88
not_in() (py42.sdk.queries.fileevents.filters.email_filter.EmailPolicyName class method), 86
not_in() (py42.sdk.queries.fileevents.filters.email_filter.EmailRecipients class method), 87
not_in() (py42.sdk.queries.fileevents.filters.email_filter.EmailSender class method), 88
not_in() (py42.sdk.queries.fileevents.filters.email_filter.EmailSubject class method), 87
not_in() (py42.sdk.queries.fileevents.filters.event_filter.EventType class method), 62
not_in() (py42.sdk.queries.fileevents.filters.event_filter.Source class method), 64
not_in() (py42.sdk.queries.fileevents.filters.exposure_filter.ExposureType class method), 77
not_in() (py42.sdk.queries.fileevents.filters.exposure_filter.ProcessName class method), 78
not_in() (py42.sdk.queries.fileevents.filters.exposure_filter.ProcessOwner class method), 79
not_in() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaMediaName class method), 81
not_in() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaName class method), 79
not_in() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaPartitionID class method), 82
not_in() (py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaSerialNumber class method), 83

not_in() (*py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaVendor class method*), 80
not_in() (*py42.sdk.queries.fileevents.filters.exposure_filter.RemovableMediaVolumeName class method*), 81
not_in() (*py42.sdk.queries.fileevents.filters.exposure_filter.SyncDestination class method*), 84
not_in() (*py42.sdk.queries.fileevents.filters.exposure_filter.TabURL class method*), 85
not_in() (*py42.sdk.queries.fileevents.filters.exposure_filter.WindowTitle class method*), 85
not_in() (*py42.sdk.queries.fileevents.filters.file_filter.FileCategory class method*), 65
not_in() (*py42.sdk.queries.fileevents.filters.file_filter.FileName class method*), 66
not_in() (*py42.sdk.queries.fileevents.filters.file_filter.FileOwner class method*), 67
not_in() (*py42.sdk.queries.fileevents.filters.file_filter.FilePath class method*), 67
not_in() (*py42.sdk.queries.fileevents.filters.file_filter.MD5 class method*), 68
not_in() (*py42.sdk.queries.fileevents.filters.file_filter.SHA256 class method*), 69
not_in() (*py42.sdk.queries.fileevents.filters.print_filter.Printer class method*), 90
not_in() (*py42.sdk.queries.fileevents.filters.print_filter.PrintJobName class method*), 90
not_in() (*py42.sdk.queries.query_filter.QueryFilterStringField class method*), 55
notes (*py42.clients.settings.device_settings.DeviceSettings attribute*), 30
notes (*py42.clients.settings.org_settings.OrgSettings attribute*), 22

O

on_or_after() (*py42.sdk.queries.alerts.filters.alert_filter.DateObserved class method*), 44
on_or_after() (*py42.sdk.queries.fileevents.filters.event_filter.EventTimestamp class method*), 60
on_or_after() (*py42.sdk.queries.fileevents.filters.event_filter.InsertionTimestamp class method*), 62
on_or_after() (*py42.sdk.queries.query_filter.QueryFilterTimestampField class method*), 55
on_or_before() (*py42.sdk.queries.alerts.filters.alert_filter.DateObserved class method*), 45
on_or_before() (*py42.sdk.queries.fileevents.filters.event_filter.EventTimestamp class method*), 60
on_or_before() (*py42.sdk.queries.fileevents.filters.event_filter.InsertionTimestamp class method*), 62
on_or_before() (*py42.sdk.queries.query_filter.QueryFilterTimestampField class method*), 55
on_same_day() (*py42.sdk.queries.alerts.filters.alert_filter.DateObserved class method*), 45
on_same_day() (*py42.sdk.queries.fileevents.filters.event_filter.EventTimestamp class method*), 61
on_same_day() (*py42.sdk.queries.fileevents.filters.event_filter.InsertionTimestamp class method*), 62
on_same_day() (*py42.sdk.queries.query_filter.QueryFilterTimestampField class method*), 55
operator (*py42.sdk.queries.query_filter.QueryFilter attribute*), 54
org_block_quota (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
org_id (*py42.clients.settings.device_settings.DeviceSettings attribute*), 30
org_type (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
org_uid (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
orgs (*py42.sdk.SDKClient attribute*), 107
OrgService (*class in py42.services.orgs*), 18
OrgSettings (*class in py42.clients.settings.org_settings*), 21
OSHostname (*class in py42.sdk.queries.fileevents.filters.device_filter*), 70
OutsideActiveHours (*class in py42.sdk.queries.fileevents.filters.event_filter*), 64

P

packets (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
plan_uid (*py42.clients.securitydata.PlanStorageInfo attribute*), 34
PlanStorageInfo (*class in py42.clients.securitydata*), 33
print_response() (*in module py42.util*), 106
Printer (*class in py42.sdk.queries.fileevents.filters.print_filter*), 89
PrintJobName (*class in py42.sdk.queries.fileevents.filters.print_filter*), 90
PrivateIPAddress (*class in py42.sdk.queries.fileevents.filters.device_filter*), 71
ProcessName (*class in py42.sdk.queries.fileevents.filters.exposure_filter*), 77
ProcessOwner (*class in py42.sdk.queries.fileevents.filters.exposure_filter*), 78
PublicIPAddress (*class in py42.sdk.queries.fileevents.filters.device_filter*), 71
py42.exceptions (*module*), 99
py42.sdk (*module*), 106
py42.sdk.queries.alerts.filters.alert_filter (*module*), 42

py42.sdk.queries.query_filter (*module*), 53
py42.util (*module*), 105
Py42ActiveLegalHoldError, 99
Py42ArchiveFileNotFoundError, 99
Py42BadRequestError, 100
Py42BadRestoreRequestError, 100
Py42CaseAlreadyHasEventError, 100
Py42CaseNameExistsError, 100
Py42ChecksumNotFoundError, 100
Py42CloudAliasLimitExceededError, 101
Py42DescriptionLimitExceededError, 101
Py42DeviceNotConnectedError, 101
Py42Error, 101
Py42FeatureUnavailableError, 101
Py42ForbiddenError, 101
Py42HTTPError, 102
Py42InternalServerError, 102
Py42InvalidArchiveEncryptionKey, 102
Py42InvalidArchivePassword, 102
Py42InvalidPageTokenError, 102
Py42InvalidRuleError, 102
Py42InvalidRuleOperationError, 103
Py42LegalHoldCriteriaMissingError, 103
Py42LegalHoldNotFoundOrPermissionDeniedError, 103
Py42MFARequiredError, 103
Py42NotFoundError, 103
Py42Response (*class in py42.response*), 99
Py42ResponseError, 103
Py42SecurityPlanConnectionError, 104
Py42SessionInitializationError, 104
Py42StorageSessionInitializationError, 104
Py42TooManyRequestsError, 104
Py42UnauthorizedError, 104
Py42UpdateClosedCaseError, 104
Py42UserAlreadyAddedError, 105
Py42UserAlreadyExistsError, 105
Py42UserNotOnListError, 105

Q

QueryFilter (*class in py42.sdk.queries.query_filter*), 54
QueryFilterBooleanField (*class in py42.sdk.queries.query_filter*), 54
QueryFilterStringField (*class in py42.sdk.queries.query_filter*), 54
QueryFilterTimestampField (*class in py42.sdk.queries.query_filter*), 55
quota_settings_inherited (*py42.clients.settings.org_settings.OrgSettings attribute*), 22

R

raise_py42_error() (*in module py42.exceptions*), 105
raw_text (*py42.response.Py42Response attribute*), 99
reactivate() (*py42.services.devices.DeviceService method*), 29
reactivate() (*py42.services.orgs.OrgService method*), 20
reactivate() (*py42.services.users.UserService method*), 26
reactivate_matter() (*py42.services.legalhold.LegalHoldService method*), 37
refresh_user_scim_attributes() (*py42.clients.detectionlists.DetectionListsClient method*), 38
registration_key (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
RemoteActivity (*class in py42.sdk.queries.fileevents.filters.activity_filter*), 89
RemovableMediaMediaName (*class in py42.sdk.queries.fileevents.filters.exposure_filter*), 80
RemovableMediaName (*class in py42.sdk.queries.fileevents.filters.exposure_filter*), 79
RemovableMediaPartitionID (*class in py42.sdk.queries.fileevents.filters.exposure_filter*), 82
RemovableMediaSerialNumber (*class in py42.sdk.queries.fileevents.filters.exposure_filter*), 82
RemovableMediaVendor (*class in py42.sdk.queries.fileevents.filters.exposure_filter*), 79
RemovableMediaVolumeName (*class in py42.sdk.queries.fileevents.filters.exposure_filter*), 81
remove() (*py42.services.detectionlists.departing_employee.DepartingEm method*), 40
remove() (*py42.services.detectionlists.high_risk_employee.HighRiskEmp method*), 41
remove_all_users() (*py42.clients.alertrules.AlertRulesClient method*), 52
remove_destination() (*py42.clients.settings.device_settings.BackupSet method*), 30
remove_from_matter() (*py42.services.legalhold.LegalHoldService method*), 37
remove_role() (*py42.services.users.UserService method*), 26

remove_user() (*py42.clients.alertrules.AlertRulesClient method*), 52
remove_user_cloud_alias() (*py42.clients.detectionlists.DetectionListsClient method*), 38
remove_user_risk_tags() (*py42.clients.detectionlists.DetectionListsClient method*), 38
reopen() (*py42.clients.alerts.AlertsClient method*), 50
reporting_settings_inherited (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
resolve() (*py42.clients.alerts.AlertsClient method*), 50
response (*py42.exceptions.Py42ActiveLegalHoldError attribute*), 99
response (*py42.exceptions.Py42ArchiveFileNotFoundError attribute*), 100
response (*py42.exceptions.Py42BadRequestError attribute*), 100
response (*py42.exceptions.Py42BadRestoreRequestError attribute*), 100
response (*py42.exceptions.Py42CaseAlreadyHasEventError attribute*), 100
response (*py42.exceptions.Py42CaseNameExistsError attribute*), 100
response (*py42.exceptions.Py42ChecksumNotFoundError attribute*), 100
response (*py42.exceptions.Py42CloudAliasLimitExceededError attribute*), 101
response (*py42.exceptions.Py42DescriptionLimitExceededError attribute*), 101
response (*py42.exceptions.Py42DeviceNotConnectedError attribute*), 101
response (*py42.exceptions.Py42FeatureUnavailableError attribute*), 101
response (*py42.exceptions.Py42ForbiddenError attribute*), 101
response (*py42.exceptions.Py42HTTPError attribute*), 102
response (*py42.exceptions.Py42InternalServerError attribute*), 102
response (*py42.exceptions.Py42InvalidArchiveEncryptionKey attribute*), 102
response (*py42.exceptions.Py42InvalidArchivePassword attribute*), 102
response (*py42.exceptions.Py42InvalidPageTokenError attribute*), 102
response (*py42.exceptions.Py42InvalidRuleError attribute*), 102
response (*py42.exceptions.Py42InvalidRuleOperationError attribute*), 103
response (*py42.exceptions.Py42LegalHoldCriteriaMissingError attribute*), 103
response (*py42.exceptions.Py42LegalHoldNotFoundOrPermissionDenie attribute*), 103
response (*py42.exceptions.Py42MFARequiredError attribute*), 103
response (*py42.exceptions.Py42NotFoundError attribute*), 103
response (*py42.exceptions.Py42ResponseError attribute*), 103
response (*py42.exceptions.Py42SecurityPlanConnectionError attribute*), 104
response (*py42.exceptions.Py42StorageSessionInitializationError attribute*), 104
response (*py42.exceptions.Py42TooManyRequestsError attribute*), 104
response (*py42.exceptions.Py42UnauthorizedError attribute*), 104
response (*py42.exceptions.Py42UpdateClosedCaseError attribute*), 105
response (*py42.exceptions.Py42UserAlreadyAddedError attribute*), 105
response (*py42.exceptions.Py42UserAlreadyExistsError attribute*), 105
response (*py42.exceptions.Py42UserNotOnListError attribute*), 105
RuleId (*class in py42.sdk.queries.alerts.filters.alert_filter*), 46
RuleName (*class in py42.sdk.queries.alerts.filters.alert_filter*), 46
rules (*py42.clients.alerts.AlertsClient attribute*), 50
RuleSource (*class in py42.sdk.queries.alerts.filters.alert_filter*), 47
RuleType (*class in py42.sdk.queries.alerts.filters.alert_filter*), 48

S

savedsearches (*py42.clients.securitydata.SecurityDataClient attribute*), 33
SavedSearchService (*class in py42.services.savedsearch*), 58
SDKClient (*class in py42.sdk*), 107
search() (*py42.clients.alerts.AlertsClient method*), 50
search_all_file_events() (*py42.clients.securitydata.SecurityDataClient method*), 33
search_file_events() (*py42.clients.securitydata.SecurityDataClient method*), 33
securitydata (*py42.sdk.SDKClient attribute*), 108
SecurityDataClient (*class in py42.clients.securitydata*), 31
serveradmin (*py42.sdk.SDKClient attribute*), 108
set_alerts_enabled() (*py42.services.detectionlists.departing_employee.DepartingEmpl*

method), 40
set_alerts_enabled() (*py42.services.detectionlists.high_risk_employee.HighRiskEmployeeService method*), 41
Severity (*class in py42.sdk.queries.alerts.filters.alert_filter*), 48
SHA256 (*class in py42.sdk.queries.fileevents.filters.file_filter*), 69
Shared (*class in py42.sdk.queries.fileevents.filters.cloud_filter*), 74
SharedWith (*class in py42.sdk.queries.fileevents.filters.cloud_filter*), 75
SharingTypeAdded (*class in py42.sdk.queries.fileevents.filters.cloud_filter*), 75
SortDirection (*class in py42.constants*), 106
Source (*class in py42.sdk.queries.fileevents.filters.event_filter*), 63
status_code (*py42.response.Py42Response attribute*), 99
stream_file_by_md5() (*py42.clients.securitydata.SecurityDataClient method*), 33
stream_file_by_sha256() (*py42.clients.securitydata.SecurityDataClient method*), 33
stream_from_backup() (*py42.clients.archive.ArchiveClient method*), 92
stream_to_device() (*py42.clients.archive.ArchiveClient method*), 93
SyncDestination (*class in py42.sdk.queries.fileevents.filters.exposure_filter*), 83

T

TabURL (*class in py42.sdk.queries.fileevents.filters.exposure_filter*), 84
term (*py42.sdk.queries.query_filter.QueryFilter attribute*), 54
text (*py42.response.Py42Response attribute*), 99
TrustedActivity (*class in py42.sdk.queries.fileevents.filters.activity_filter*), 88

U

unblock() (*py42.services.devices.DeviceService method*), 29
unblock() (*py42.services.orgs.OrgService method*), 20
unblock() (*py42.services.users.UserService method*), 26
unlock_destination() (*py42.clients.settings.device_settings.BackupSet method*), 21
update() (*py42.clients.cases.CasesClient method*), 98
update_cold_storage_purge_date() (*py42.clients.archive.ArchiveClient method*), 94
update_departure_date() (*py42.services.detectionlists.departing_employee.DepartingEmployeeService method*), 40
update_settings() (*py42.services.devices.DeviceService method*), 29
update_settings() (*py42.services.orgs.OrgService method*), 20
update_state() (*py42.clients.alerts.AlertsClient method*), 51
update_user_notes() (*py42.clients.detectionlists.DetectionListsClient method*), 38
url (*py42.response.Py42Response attribute*), 99
user_backup_quota (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
user_id (*py42.clients.settings.device_settings.DeviceSettings attribute*), 30
UserContext (*class in py42.usercontext*), 26
usercontext (*py42.sdk.SDKClient attribute*), 108
users (*py42.sdk.SDKClient attribute*), 108
UserService (*class in py42.services.users*), 23

V

value (*py42.sdk.queries.query_filter.QueryFilter attribute*), 54
version (*py42.clients.settings.device_settings.DeviceSettings attribute*), 30

W

warning_alert_days (*py42.clients.settings.device_settings.DeviceSettingsDefaults attribute*), 23
warning_email_enabled (*py42.clients.settings.device_settings.DeviceSettingsDefaults attribute*), 23
web_restore_admin_limit (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
web_restore_enabled (*py42.clients.settings.org_settings.OrgSettings attribute*), 22
web_restore_user_limit (*py42.clients.settings.org_settings.OrgSettings attribute*), 22

WindowTitle (*class in py42.sdk.queries.fileevents.filters.exposure_filter*), 85
with_traceback() (*py42.exceptions.Py42ActiveLegalHoldError method*), 99
with_traceback() (*py42.exceptions.Py42ArchiveFileNotFoundError method*), 100
with_traceback() (*py42.exceptions.Py42BadRequestError method*), 100
with_traceback() (*py42.exceptions.Py42BadRestoreRequestError method*), 100
with_traceback() (*py42.exceptions.Py42CaseAlreadyHasEventError method*), 100
with_traceback() (*py42.exceptions.Py42CaseNameExistsError method*), 100
with_traceback() (*py42.exceptions.Py42ChecksumNotFoundError method*), 100
with_traceback() (*py42.exceptions.Py42CloudAliasLimitExceededError method*), 101
with_traceback() (*py42.exceptions.Py42DescriptionLimitExceededError method*), 101
with_traceback() (*py42.exceptions.Py42DeviceNotConnectedError method*), 101
with_traceback() (*py42.exceptions.Py42Error method*), 101
with_traceback() (*py42.exceptions.Py42FeatureUnavailableError method*), 101
with_traceback() (*py42.exceptions.Py42ForbiddenError method*), 101
with_traceback() (*py42.exceptions.Py42HTTPError method*), 102
with_traceback() (*py42.exceptions.Py42InternalServerError method*), 102
with_traceback() (*py42.exceptions.Py42InvalidArchiveEncryptionKey method*), 102
with_traceback() (*py42.exceptions.Py42InvalidArchivePassword method*), 102
with_traceback() (*py42.exceptions.Py42InvalidPageTokenError method*), 102
with_traceback() (*py42.exceptions.Py42InvalidRuleError method*), 102
with_traceback() (*py42.exceptions.Py42InvalidRuleOperationError method*), 103
with_traceback() (*py42.exceptions.Py42LegalHoldCriteriaMissingError method*), 103
with_traceback() (*py42.exceptions.Py42LegalHoldNotFoundOrPermissionDeniedError method*), 103
with_traceback() (*py42.exceptions.Py42MFARequiredError method*), 103
with_traceback() (*py42.exceptions.Py42NotFoundError method*), 103
with_traceback() (*py42.exceptions.Py42ResponseError method*), 104
with_traceback() (*py42.exceptions.Py42SecurityPlanConnectionError method*), 104
with_traceback() (*py42.exceptions.Py42SessionInitializationError method*), 104
with_traceback() (*py42.exceptions.Py42StorageSessionInitializationError method*), 104
with_traceback() (*py42.exceptions.Py42TooManyRequestsError method*), 104
with_traceback() (*py42.exceptions.Py42UnauthorizedError method*), 104
with_traceback() (*py42.exceptions.Py42UpdateClosedCaseError method*), 105
with_traceback() (*py42.exceptions.Py42UserAlreadyAddedError method*), 105
with_traceback() (*py42.exceptions.Py42UserAlreadyExistsError method*), 105
with_traceback() (*py42.exceptions.Py42UserNotOnListError method*), 105
within_the_last() (*py42.sdk.queries.fileevents.filters.event_filter.EventTimestamp class method*), 61
within_the_last() (*py42.sdk.queries.fileevents.filters.event_filter.InsertionTimestamp class method*), 63