

# ThreatConnect™ Java API

## *Quick Start Guide*

DRAFT

# Table of Contents

[About This Document](#)

[How to Use This Document](#)

[Getting Started](#)

[Technical Design](#)

[First API Example](#)

[The Reader Package](#)

[Reader Factory](#)

[Reader Factory Example](#)

[Reader Class Overview](#)

[Parameter Naming Convention](#)

[AbstractGroupReaderAdapter Class](#)

[Associated Groups](#)

[Associated Indicators](#)

[Associated Security Labels](#)

[Associated Tags](#)

[Associated VictimAssets](#)

[Associated Attributes](#)

[AbstractIndicatorReaderAdapter Class](#)

[OwnerReaderAdapter Class](#)

[SecurityLabelReaderAdapter Class](#)

[TagReaderAdapter Class](#)

[VictimReaderAdapter Class](#)

[Reader IP Address and Tag Example](#)

## [Reader Example 2](#)

### [Writer Package](#)

#### [Writer Factory](#)

#### [Writer Responses](#)

#### [Writer Create Example](#)

#### [AbstractGroupWriterAdapter Class](#)

##### [Associate Groups](#)

##### [Associate Indicators](#)

##### [Associate Security Labels](#)

##### [Associate Tag](#)

##### [Associate Victim](#)

##### [Associate Victim Asset](#)

##### [Add Attributes](#)

##### [Update Attribute](#)

##### [Delete Group Association](#)

##### [Delete Indicator Associations](#)

##### [Delete Security Label Associations](#)

##### [Delete Tag Associations](#)

##### [Delete Victim Associations](#)

##### [Delete VictimAsset Associations](#)

##### [Delete Attribute](#)

##### [Update Attribute](#)

#### [AbstractIndicatorWriterAdapter](#)

##### [SecurityLabelWriterAdapter](#)

##### [TagWriterAdapter](#)

## [VictimWriterAdapter](#)

### [Writer Examples](#)

[Writer Delete Example](#)

[Writer Update Example](#)

[Writer Add Attribute Example](#)

[Writer Associate Indicator Example](#)

[Writer Associate Group Example](#)

[Writer Associate Tag Example](#)

[Writer Associate Victim Example](#)

[Writer Remove Association Example](#)

## About This Document

This quick start guide gets you started coding Java applications using the ThreatConnect API. The Java API offers coverage of all features in version 2.0 of the ThreatConnect API -- this includes the ability to write data to ThreatConnect. This document will provide an overview of the reference implementation of the ThreatConnect Java API.

The goal of this Java API library is to provide a programmatic abstraction layer around the ThreatConnect API without losing functional coverage over the available API resources. This abstraction layer enables developers to focus on writing enterprise functionality without worrying about low-level RESTful calls and authentication management.



This document is not a replacement for the official ThreatConnect API Documentation. This document serves as a companion to the official documentation for the RESTful ThreatConnect API. Read the official documentation to gain a further understanding of the functional aspects of using the ThreatConnect API.

## How to Use This Document

This document will teach you how to create groups, indicators, associations, tags, security labels, and victims. Along with creating data elements, you will be able to create, update, delete, and request data from the API using Java. This document assumes the reader knows the Java Programming Language.

All code examples will be noted in a separate box with a monospaced font and line numbers to facilitate explanation of code functionality. When a single line of code wraps, the rounded right arrow “↪” will highlight that the code is a continuation of the prior line (see between line 2 and 3 below). This is a code sample with line numbers and syntax highlighting.

```
1  private static void doCreate(Connection conn) {
2      AbstractGroupWriterAdapter<Adversary> writer
↪  WriterAdapterFactory.createAdversaryGroupWriter(conn);
3
4      Adversary adversary = new Adversary();
5      adversary.setName("Test Adversary");
6      adversary.setOwnerName("System");
7
8      try {
9          Adversary savedAdversary = writer.create(adversary);
10         System.out.println("Saved: " + savedAdversary.toString() );
11     } catch (IOException | FailedResponseException ex) {
12         System.err.println("Error: " + ex.toString());
13     }
14
15
16 }
```

## Getting Started

To get started, you'll need to have JDK 7+ installed along with the ThreatConnect Java API library. This section will also highlight basic API user configuration to connect to the ThreatConnect API. While an IDE will facilitate development of larger scale systems, it is not required to follow the examples in this document.

**Important:** The following software is required to use the ThreatConnect Java API.

- Java JDK 7+ <http://www.oracle.com/technetwork/java/javase/downloads/index.html>
- ThreatConnect Java API<Insert Download Link Here or note that it was downloaded with doc>

To use the ThreatConnect RESTful API, an API user must be provisioned. See the official ThreatConnect API documentation for details on how to create an API user as it is out of scope for this document.

The Java API will need to be configured with an Access ID and Secret Key. The Java API searches system properties for the "threatconnect.api.config" property. In Java this property can be defined in two ways:

1. In the JVM, you can call your program with the following -D property flag:

```
threatconnect.api.config=<YOUR CONFIG FILE LOCATION>
```

2. The system property can be directly set at runtime using the following code:

```
System.getProperties().setProperty("threatconnect.api.config", "<YOUR CONFIG FILE LOCATION>");
```

The configuration file should contain the following lines at a minimum:

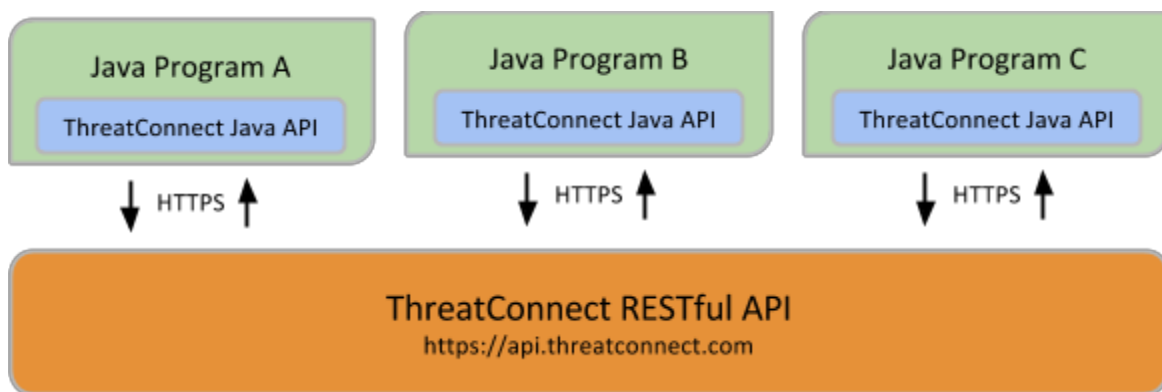
```
connection.tcApiUrl=https://api.threatconnect.com
connection.tcApiAccessID=<YOUR API ACCESS ID>
connection.tcApiUserSecretKey=<YOUR API SECRET KEY>
```

Once the configuration has been set up, you should be able to run the examples in this document as long as the ThreatConnect Java API is part of your classpath. See the following examples for a typical startup script.

```
Windows: java -cp ".;tcApiLib.jar" -Dthreatconnect.api.config=myConfig.properties TestClass
*nix:    java -cp " ../tcApiLib.jar" -Dthreatconnect.api.config=myConfig.properties TestClass
```

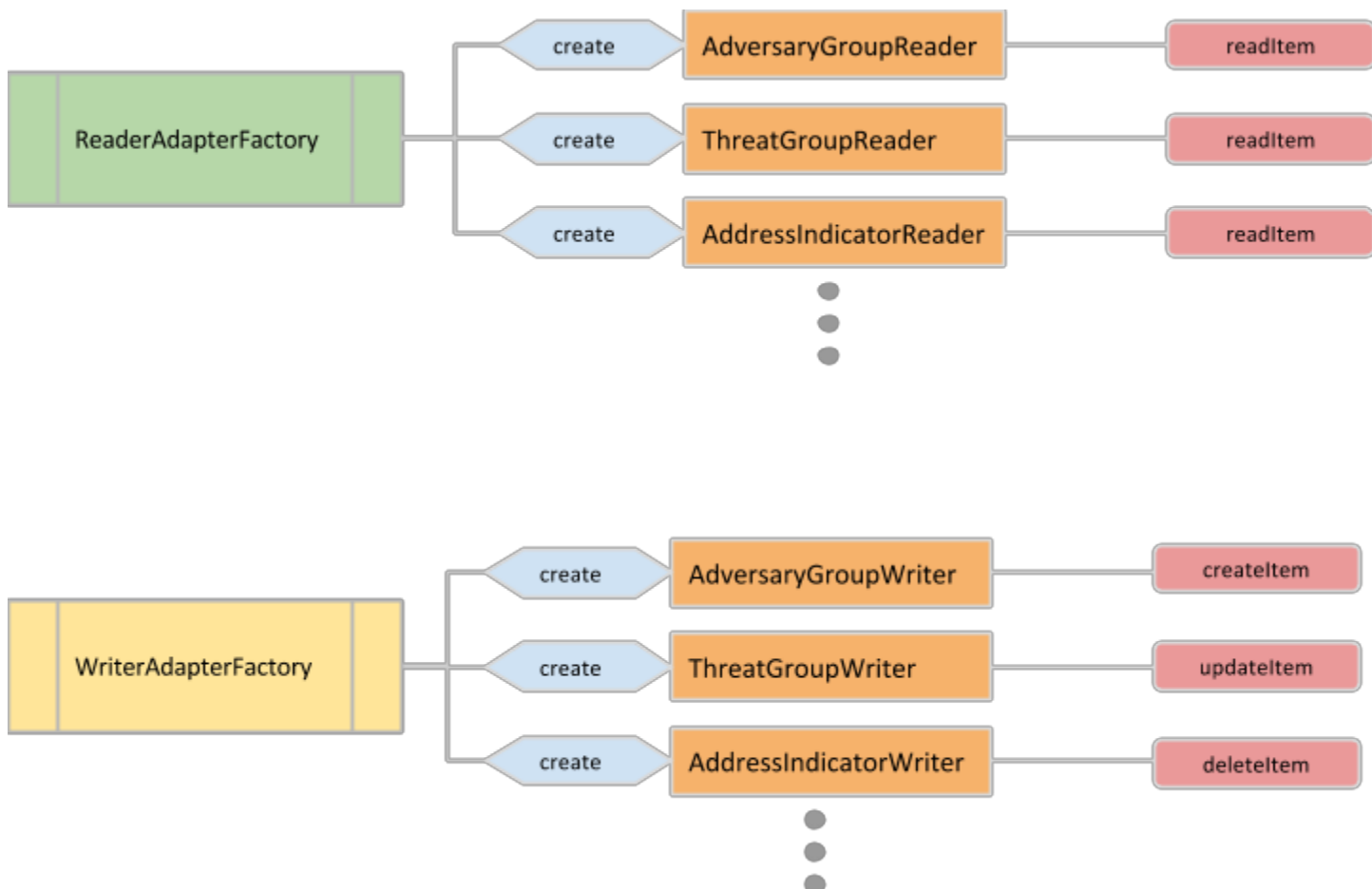
## Technical Design

The ThreatConnect Java API was designed with a focus on abstracting the API REST calls while enabling the developer to use an enterprise level programming language. The abstraction layer is relatively “thin” because it coincides directly with all of the REST API calls. In fact, the entities themselves were ported directly from the ThreatConnect API to enable consistent communication between the Java API and the REST API.



The Java library was designed with common programming design patterns. You’ll notice the “Adapter” pattern used to manage the interaction with the API connection and REST calls. The Java API depends the Apache HTTP Components open source library to handle these calls. Because instantiating an Adapter requires a low-level RequestExecutor, a “Factory” design pattern was utilized to expose reading/writing functionality in a simplified way.

You’ll notice the use of Java Generics to type many of the Adapters in an effort to reuse code, as most readers share functional resources. Below is a diagram that will help illustrate common interactions between different classes (Note: names are conceptual to illustrate interaction, actual class names and methods will be discussed later in this document) . All interactions with the Java API will follow this programmatic idiom.



To facilitate interaction with the full set of Java API readers and writers, it's highly advised to use the ReaderAdapterFactory and WriterAdapterFactory respectively.

## First API Example

Now that we've covered setup and the Java API design, let's write our first program using the Java API. We'll create an Adversary reader to pull a collection of all Adversaries belonging to organization "System". Once retrieved, the adversary objects will be printed to the console.

```
1 import com.cyber2.api.lib.client.reader.AbstractGroupReaderAdapter;
2 import com.cyber2.api.lib.client.reader.ReaderAdapterFactory;
3 import com.cyber2.api.lib.conn.Connection;
4 import com.cyber2.api.lib.exception.FailedResponseException;
5 import com.cyber2.api.lib.server.entity.Adversary;
6 import java.io.IOException;
7 import java.util.List;
```



```

8
9 public class GroupExample {
10
11     public static void main(String[] args) {
12
13         Connection conn = null;
14
15         try {
16
17             System.getProperties().setProperty("threatconnect.api.config", "/config.properties");
18             conn = new Connection();
19
20             AbstractGroupReaderAdapter<Adversary> reader =
↪ ReaderAdapterFactory.createAdversaryGroupReader(conn);
21             List<Adversary> data = reader.getAll("System");
22             for (Adversary g : data ) {
23                 System.out.println( "Adversary: " + g.toString() );
24             }
25
26         } catch (IOException | FailedResponseException ex) {
27             System.err.println("Error: " + ex);
28         } finally {
29             if ( conn != null )      conn.disconnect();
30         }
31     }
32 }
33
34 }

```

- Line 1-7      Notable imports include:
- ➔ The “com.cyber2.api.lib.client.reader” package holds all Adapter classes that read data from the API
  - ➔ The “com.cyber2.api.lib.server.entity” package holds all entities returned by the Java API
- Line 17-18      We programmatically define the system property to load the configuration file. This allows the developer to instantiate Connection objects (line 18) with a no-arg constructor. If the “threatconnect.api.config” property isn’t defined, the developer has the option of passing the configuration file name string in the single-arg Connection constructor.
- Line 20      Here is the standard way to create an AbstractGroupReaderAdapter<Adversary> object. Using the ReaderAdapterFactory pattern and generics, we enforce compile-time type constraints on this abstract class. We pass the connection object used by the Adapter to interact with the ThreatConnect API.
- Line 21      Using the reader object, we call getAll() method and pass it the organization string name to return all Adversaries for the “System” organization.
- Line 22-24      We iterate through the data collection to print the contents to the console.
- Line 26      The IOException is potentially thrown if the Connection object can’t find the properties file. The FailedResponseException is thrown if the API request is invalid.

Line 29            In all cases when processing is complete, we call `disconnect()` on the connection object to release resources.

In this section we learned:

- How to connect to the ThreatConnect API by passing the configuration file in system properties
- How to get a list of Adversaries for organization “System”
- What types of exceptions a connection and read operation can potentially throw
- How to close a ThreatConnect API connection

## The Reader Package

The reader package is the primary package to retrieve data from the ThreatConnect API. It covers all available resources exposed through the ThreatConnect API. The primary classes in the reader package are listed below. They encompass all read functionality from the ThreatConnect API.

Class	Description
<code>ReaderAdapterFactory</code>	Primary entry point to instantiate all readers in the reader package.
<code>AbstractGroupReaderAdapter&lt;T extends Group&gt;</code>	Generic Group Reader Abstract class. Concrete object available in <code>ReaderAdapterFactory</code> .
<code>AbstractIndicatorReaderAdapter&lt;T extends Indicator&gt;</code>	Generic Indicator Reader Abstract class. Concrete object available in <code>ReaderAdapterFactory</code> .
<code>AbstractReaderAdapter</code>	Base Abstract Reader for all Reader Adapters in the reader package.
<code>OwnerReaderAdapter</code>	Concrete Reader for Organization owner data. Convenience object available in <code>ReaderAdapterFactory</code> .
<code>SecurityLabelReaderAdapter</code>	Concrete Reader for SecurityLabel data. Convenience object available in <code>ReaderAdapterFactory</code> .
<code>TagReaderAdapter</code>	Concrete Reader for Tag data. Convenience object available in <code>ReaderAdapterFactory</code> .
<code>VictimReaderAdapter</code>	Concrete Reader for Victim data. Convenience object available in <code>ReaderAdapterFactory</code> .

## Reader Factory

The ReaderAdapterFactory class is effectively the “hub” for reader adapters. It provides convenience objects for all the adapters in the reader package. Below is a list of the static methods and return types of the **ReaderAdapterFactory**.

Modifier and Type	Method
static AbstractGroupReaderAdapter<Adversary>	createAdversaryGroupReader(Connection conn)
static AbstractGroupReaderAdapter<Email>	createEmailGroupReader(Connection conn)
static AbstractGroupReaderAdapter<Incident>	createIncidentGroupReader(Connection conn)
static AbstractGroupReaderAdapter<Signature>	createSignatureGroupReader(Connection conn)
static AbstractGroupReaderAdapter<Threat>	createThreatGroupReader(Connection conn)
static AbstractIndicatorReaderAdapter<Address>	createAddressIndicatorReader(Connection conn)
static AbstractIndicatorReaderAdapter<EmailAddress>	createEmailAddressIndicatorReader(Connection conn)
static AbstractIndicatorReaderAdapter<File>	createFileIndicatorReader(Connection conn)
static AbstractIndicatorReaderAdapter<Host>	createHostIndicatorReader(Connection conn)
static AbstractIndicatorReaderAdapter<Url>	createUrlIndicatorReader(Connection conn)
static OwnerReaderAdapter	createOwnerReader(Connection conn)
static SecurityLabelReaderAdapter	createSecurityLabelReader(Connection conn)
static TagReaderAdapter	createTagReader(Connection conn)
static VictimReaderAdapter	createVictimReader(Connection conn)

## Reader Factory Example

Let's continue building from our first example and use more Adapters available to us in the reader package. The following example reads all groups available to the "System" organization. It then proceeds to iterate through each group printing and performing "getId()" lookups to get the full Group object from the ThreatConnect API. Note the "..." characters were added to code sections removed for brevity.

```
1 import com.cyber2.api.lib.client.reader.AbstractGroupReaderAdapter;
2 import com.cyber2.api.lib.client.reader.ReaderAdapterFactory;
3 import com.cyber2.api.lib.conn.Connection;
4 import com.cyber2.api.lib.exception.FailedResponseException;
5 import com.cyber2.api.lib.server.entity.Adversary;
6 import com.cyber2.api.lib.server.entity.Email;
7 import com.cyber2.api.lib.server.entity.Group;
8 import com.cyber2.api.lib.server.entity.Incident;
9 import com.cyber2.api.lib.server.entity.Signature;
10 import com.cyber2.api.lib.server.entity.Threat;
11 import java.io.IOException;
12 import java.util.List;
13
14 ...
53 private static void doGetById(Connection conn) throws IOException, FailedResponseException {
54
55     AbstractGroupReaderAdapter reader = ReaderAdapterFactory.createAdversaryGroupReader(conn);
56     List<Group> data = reader.getAllGroups();
57     for (Group group : data) {
58         System.err.println("Checking group.class=" + group.getClass() + ", type=" + group.getType());
59         Group result = null;
60         switch( Group.Type.valueOf(group.getType()) ) {
61             case Adversary:
62                 AbstractGroupReaderAdapter<Adversary> adversaryReader
63                     = ReaderAdapterFactory.createAdversaryGroupReader(conn);
64                 // "result" is assigned an Adversary object
65                 result = adversaryReader.getById(group.getId(), group.getOwnerName());
66                 break;
67             case Email:
68                 AbstractGroupReaderAdapter<Email> emailReader
69                     = ReaderAdapterFactory.createEmailGroupReader(conn);
70                 // "result" is assigned an Email object
71                 result = emailReader.getById(group.getId(), group.getOwnerName());
72                 break;
73             case Incident:
74                 AbstractGroupReaderAdapter<Incident> incidentReader
75                     = ReaderAdapterFactory.createIncidentGroupReader(conn);
76                 // "result" is assigned an Incident object
77                 result = incidentReader.getById(group.getId(), group.getOwnerName() );
78                 break;
79             case Signature:
80                 AbstractGroupReaderAdapter<Signature> sigReader
81                     = ReaderAdapterFactory.createSignatureGroupReader(conn);
82                 // "result" is assigned a Signature object
83                 result = sigReader.getById(group.getId(), group.getOwnerName() );
84                 break;
85             case Threat:
86                 AbstractGroupReaderAdapter<Threat> threatReader
87                     = ReaderAdapterFactory.createThreatGroupReader(conn);
88                 // "result" is assigned a Threat object
```

```

89         result = threatReader.getById(group.getId(), group.getOwnerName() );
90         break;
91     default:
92         System.err.println("Unknown Group Type: " + group.getType() );
93         break;
94     }
95
96     assert result.getId().equals(group.getId());
97 }
98
99 }

```

- Lines 5-10      Notice how we've added all Group level entities in the imports. Results from reader adapters will return an entity or a collection of entities from the "com.cyber2.api.lib.server.entity" package.
- Line 52-53    We are interested in retrieving all groups that the current API user has access to under the "System" organization. All AbstractGroupReaderAdapter's have access to the "getAllGroups()" method -- it returns a collection of Group objects for the "System" organization from the ThreatConnect API.
- Line 60        To illustrate the different instantiations, we use a switch statement on the generic group object.
- Line 61-63    Based on the Group.Type enum value, in this section "Adversary", we create an Adversary Group Reader object from the ReaderAdapterFactory. The assignment to the adversaryReader variable is typed using generics to enforce compile time checks on the data returned from this reader.
- Line 65        Here we use the "getById()" method to retrieve the proper Adversary group data, based on the ID and organization name, from the ThreatConnect API. The "result" variable is assigned an Adversary type object.
- Line 67-90    The remaining case statement blocks check for different Group types, but effectively do the same operation. Take some time to review these blocks to understand how the ReaderFactory facilitates the creation of proper readers.
- Line 96        Here we compare the group ID against the result ID returned by the "getById" method to assert that they are in fact the same entity.



There are more concise ways to handle reading data and purely checking it's ID. This code was written in a more verbose form strictly to illustrate the usage of different methods in the ReaderFactory.

## Reader Class Overview

While the main entry point to the reader package is the ReaderFactory, getting familiar with the main Adapters will help you understand how to interact with the data returning from the ThreatConnect API. While there is extensive

use of Java Generics, the method naming conventions will be familiar and self-explanatory. Parameter naming conventions have been kept abstract to allow for a better representation of the identifiers being passed.

## Parameter Naming Convention

Modifier and Type	Method
uniqueId	<p>Identifier for the reader/writer Group or Incident Adapter type.</p> <p>For Groups, this is an Integer that requires an Adversary ID, Email ID, Incident ID, Signature ID, or Threat ID. This identifier is system generated when the group is created in ThreatConnect.</p> <p>For Indicators, this is a String that requires an IP Address, Email Address, File Hash, Host Name, or URL Text. This identifier is user generated when the indicator is created in ThreatConnect.</p>
victimId	Identifier for the Victim Adapter type. This identifier is an Integer created by the system when the Victim entry is created in ThreatConnect.
assetId	Identifier for the VictimAsset Adapter type. This identifier is an Integer created by the system when the VictimAsset is created in ThreatConnect. This identifier represents a VictimEmailAddress ID, VictimNetworkAccount ID, VictimPhone ID, VictimSocialNetwork ID, or VictimWebsite ID.
securityLabel	Identifier for SecurityLabel Adapter type. This is a user-provided String that represents the security label.
tagName	Identifier for Tag Adapter type. This is a user-provided String that represents the tag.

Let's start with the AbstractGroupReaderAdapter, the object returned when you call a GroupReader from the ReaderFactory. We reviewed these GroupReader instantiations in our last example.



The ThreatConnect Java API library comes with JavaDocs in the “apidocs” directory -- this is the reference for the Java API.

## AbstractGroupReaderAdapter Class

The following methods get data for the Group type (T) linked to this Adapter. The uniqueId (P) for Groups is an Integer.

Modifier and Type	Method
T	getById(P uniqueId)
T	getById(P uniqueId, String ownerName)
List<T>	getAll()
List<T>	getAll(String ownerName)

The methods below get generic Group objects associated to this Group type (T).

Modifier and Type	Method
List<Group>	getAllGroups()
List<Group>	getAllGroups(String ownerName)
String	getAllGroupsAsText()

## Associated Groups

The methods below get associated Group elements by distinct type.

Modifier and Type	Method
List<Adversary>	getAssociatedGroupAdversaries(Integer uniqueId)
List<Adversary>	getAssociatedGroupAdversaries(Integer uniqueId, String ownerName)

Adversary	getAssociatedGroupAdversary(Integer uniqueId, Integer adversaryId)
Adversary	getAssociatedGroupAdversary(Integer uniqueId, Integer adversaryId, String ownerName)
Email	getAssociatedGroupEmail(Integer uniqueId, Integer emailId)
Email	getAssociatedGroupEmail(Integer uniqueId, Integer emailId, String ownerName)
List<Email>	getAssociatedGroupEmails(Integer uniqueId)
List<Email>	getAssociatedGroupEmails(Integer uniqueId, String ownerName)
Incident	getAssociatedGroupIncident(Integer uniqueId, Integer incidentId)
Incident	getAssociatedGroupIncident(Integer uniqueId, Integer incidentId, String ownerName)
List<Incident>	getAssociatedGroupIncidents(Integer uniqueId)
List<Incident>	getAssociatedGroupIncidents(Integer uniqueId, String ownerName)
List<Group>	getAssociatedGroups(Integer uniqueId)
List<Group>	getAssociatedGroups(Integer uniqueId, String ownerName)
Signature	getAssociatedGroupSignature(Integer uniqueId, Integer signatureId)
Signature	getAssociatedGroupSignature(Integer uniqueId, Integer signatureId, String ownerName)
List<Signature>	getAssociatedGroupSignatures(Integer uniqueId)
List<Signature>	getAssociatedGroupSignatures(Integer uniqueId, String ownerName)
Threat	getAssociatedGroupThreat(Integer uniqueId, Integer threatId)
Threat	getAssociatedGroupThreat(Integer uniqueId, Integer threatId, String ownerName)
List<Threat>	getAssociatedGroupThreats(Integer uniqueId)
List<Threat>	getAssociatedGroupThreats(Integer uniqueId, String ownerName)



The following methods get associated Indicator elements by distinct types.

Modifier and Type	Method
Address	getAssociatedIndicatorAddress(Integer uniqueId, String ipAddress)
Address	getAssociatedIndicatorAddress(Integer uniqueId, String ipAddress, String ownerName)
List<Address>	getAssociatedIndicatorAddresses(Integer uniqueId)
List<Address>	getAssociatedIndicatorAddresses(Integer uniqueId, String ownerName)
Email	getAssociatedIndicatorEmail(Integer uniqueId, String emailAddress)
Email	getAssociatedIndicatorEmail(Integer uniqueId, String emailAddress, String ownerName)
List<Email>	getAssociatedIndicatorEmails(Integer uniqueId)
List<Email>	getAssociatedIndicatorEmails(Integer uniqueId, String ownerName)
File	getAssociatedIndicatorFile(Integer uniqueId, String fileHash)
File	getAssociatedIndicatorFile(Integer uniqueId, String fileHash, String ownerName)
List<File>	getAssociatedIndicatorFiles(Integer uniqueId)
List<File>	getAssociatedIndicatorFiles(Integer uniqueId, String ownerName)
Host	getAssociatedIndicatorHost(Integer uniqueId, String hostName)
Host	getAssociatedIndicatorHost(Integer uniqueId, String hostName, String ownerName)
List<Host>	getAssociatedIndicatorHosts(Integer uniqueId)
List<Host>	getAssociatedIndicatorHosts(Integer uniqueId, String ownerName)
List<Indicator>	getAssociatedIndicators(Integer uniqueId)
List<Indicator>	getAssociatedIndicators(Integer uniqueId, String ownerName)

Url	getAssociatedIndicatorUrl(Integer uniqueId, String urlText)
Url	getAssociatedIndicatorUrl(Integer uniqueId, String urlText, String ownerName)
List<Url>	getAssociatedIndicatorUrls(Integer uniqueId)
List<Url>	getAssociatedIndicatorUrls(Integer uniqueId, String ownerName)

### *Associated Security Labels*

The methods below get associated SecurityLabel data elements.

Modifier and Type	Method
SecurityLabel	getAssociatedSecurityLabel(Integer uniqueId, String securityLabelName)
SecurityLabel	getAssociatedSecurityLabel(Integer uniqueId, String securityLabelName, String ownerName)
List<SecurityLabel>	getAssociatedSecurityLabels(Integer uniqueId)
List<SecurityLabel>	getAssociatedSecurityLabels(Integer uniqueId, String ownerName)

### *Associated Tags*

The methods below get associated Tag data elements.

Modifier and Type	Method
Tag	getAssociatedTag(Integer uniqueId, String tagName)
Tag	getAssociatedTag(Integer uniqueId, String tagName, String ownerName)
List<Tag>	getAssociatedTags(Integer uniqueId)
List<Tag>	getAssociatedTags(Integer uniqueId, String ownerName)

## Associated VictimAssets

The methods below get associated VictimAsset data elements.

Modifier and Type	Method
VictimEmailAddress	getAssociatedVictimAssetEmailAddress(Integer uniqueId, Integer assetId)
VictimEmailAddress	getAssociatedVictimAssetEmailAddress(Integer uniqueId, Integer assetId, String ownerName)
List<VictimEmailAddress>	getAssociatedVictimAssetEmailAddresses(Integer uniqueId)
List<VictimEmailAddress>	getAssociatedVictimAssetEmailAddresses(Integer uniqueId, String ownerName)
VictimNetworkAccount	getAssociatedVictimAssetNetworkAccount(Integer uniqueId, Integer assetId)
VictimNetworkAccount	getAssociatedVictimAssetNetworkAccount(Integer uniqueId, Integer assetId, String ownerName)
List<VictimNetworkAccount>	getAssociatedVictimAssetNetworkAccounts(Integer uniqueId)
List<VictimNetworkAccount>	getAssociatedVictimAssetNetworkAccounts(Integer uniqueId, String ownerName)
VictimPhone	getAssociatedVictimAssetNetworkPhoneNumber(Integer uniqueId, Integer assetId)
VictimPhone	getAssociatedVictimAssetNetworkPhoneNumber(Integer uniqueId, Integer assetId, String ownerName)
List<VictimPhone>	getAssociatedVictimAssetNetworkPhoneNumbers(Integer uniqueId)
List<VictimPhone>	getAssociatedVictimAssetNetworkPhoneNumbers(Integer uniqueId, String ownerName)
List<VictimAsset>	getAssociatedVictimAssets(Integer uniqueId, String ownerName)
VictimSocialNetwork	getAssociatedVictimAssetSocialNetwork(Integer uniqueId, Integer assetId)
VictimSocialNetwork	getAssociatedVictimAssetSocialNetwork(Integer uniqueId, Integer assetId, String ownerName)
List<VictimSocialNetwork>	getAssociatedVictimAssetSocialNetworks(Integer uniqueId)
List<VictimSocialNetwork>	getAssociatedVictimAssetSocialNetworks(Integer uniqueId, String ownerName)

VictimWebSite	getAssociatedVictimAssetWebsite(Integer uniqueId, Integer assetId)
VictimWebSite	getAssociatedVictimAssetWebsite(Integer uniqueId, Integer assetId, String ownerName)
List<VictimWebSite>	getAssociatedVictimAssetWebsites(Integer uniqueId)
List<VictimWebSite>	getAssociatedVictimAssetWebsites(Integer uniqueId, String ownerName)

## Associated Attributes

The methods below Get Attributes and Attribute SecurityLabels for this Group Type.

Modifier and Type	Method
Attribute	getAttribute(Integer uniqueId, Integer attributeId)
Attribute	getAttribute(Integer uniqueId, Integer attributeId, String ownerName)
List<Attribute>	getAttributes(Integer uniqueId)
List<Attribute>	getAttributes(Integer uniqueId, String ownerName)
SecurityLabel	getAttributeSecurityLabel(Integer uniqueId, Integer attributeId, String securityLabelName)
SecurityLabel	getAttributeSecurityLabel(Integer uniqueId, Integer attributeId, String securityLabelName, String ownerName)
List<SecurityLabel>	getAttributeSecurityLabels(Integer uniqueId, Integer attributeId)
List<SecurityLabel>	getAttributeSecurityLabels(Integer uniqueId, Integer attributeId, String ownerName)

Take the time to review the different data element classifications and associations that the ThreatConnect API makes available. For the remaining Class overview in the reader package, we'll highlight the differences from the Group Adapter.

## AbstractIndicatorReaderAdapter Class

Much of the `AbstractIndicatorReaderAdapter` is similar to the `AbstractGroupReaderAdapter` Class. Indicators share the ability to associate [Groups](#), [Indicators](#), [SecurityLabels](#), [Tags](#), [VictimAssets](#), and [Attributes](#). The listings below are some distinctions or subtle differences.



All indicators in the ThreatConnect API have a uniqueId data type of “String”. This identifier is provided by each Organization in the form of an Email Address, IP Address, File Hash, Hostname, or URL text. To understand this distinction, read the Indicator section in the ThreatConnect API Documentation.

The following methods get data for the Indicator type (T) linked to this Adapter. The uniqueId (P) for Indicators is a String.

Modifier and Type	Method
T	getById(P uniqueId)
T	getById(P uniqueId, String ownerName)
List<T>	getAll()
List<T>	getAll(String ownerName)

The method below returns all the generic Indicators the current API user has access to.

Modifier and Type	Method
List<Indicator>	getIndicators()

The methods below return Owner’s who have created the indicator under the uniqueId.

Modifier and Type	Method
List<Owner>	getAssociatedOwners(String uniqueId)
List<Owner>	getAssociatedOwners(String uniqueId, String ownerName)

The `AbstractIndicatorReaderAdapter` class has a concrete subclass **`FileIndicatorReaderAdapter`** that exposes the following methods.

Modifier and Type	Method
-------------------	--------

FileOccurrence	getFileOccurrence(String uniqueId, Integer fileOccurrenceId)
FileOccurrence	getFileOccurrence(String uniqueId, Integer fileOccurrenceId, String ownerName)

## OwnerReaderAdapter Class

The OwnerReaderAdapter is a simple Adapter that returns a list of Organizations your API user has access to. There is a second method called “getOwnerMine()” that returns your Organization.

Modifier and Type	Method
Owner	getOwnerMine()
List<Owner>	getOwners()

## SecurityLabelReaderAdapter Class

The SecurityLabelReaderAdapter class is a concrete class (available through the ReaderFactory) that returns SecurityLabels your API user has access to as well as by uniqueId (P). The uniqueId data type for SecurityLabels is a String.

Modifier and Type	Method
T	getById(P uniqueId)
T	getById(P uniqueId, String ownerName)
List<T>	getAll()
List<T>	getAll(String ownerName)

In addition to retrieving basic SecurityLabel data, you can get associated [Groups](#) and [Indicators](#). For more details on those methods, look at the [AbstractGroupReaderAdapter](#) class.

## TagReaderAdapter Class

The TagReaderAdapter class is a concrete class (available through the ReaderFactory) that returns Tags your API user has access to as well as by uniqueId (P). The uniqueId data type for Tags is a String.

Modifier and Type	Method
T	getById(P uniqueId)
T	getById(P uniqueId, String ownerName)
List<T>	getAll()
List<T>	getAll(String ownerName)

In addition to retrieving basic Tag data, you can get associated [Groups](#) and [Indicators](#). For more details on those methods, review the [AbstractGroupReaderAdapter](#) class.

## VictimReaderAdapter Class

The VictimReaderAdapter class is a concrete class (available through the ReaderFactory) that returns Victims your API user has access to as well as by uniqueId (P). The uniqueId data type for a Victim is an Integer.

Modifier and Type	Method
T	getById(P uniqueId)
T	getById(P uniqueId, String ownerName)
List<T>	getAll()
List<T>	getAll(String ownerName)

In addition to retrieving basic Victim data, you can get associated [Groups](#), [Indicators](#), and [VictimAssets](#). For more details on those methods, review the [AbstractGroupReaderAdapter](#) class.

## Reader IP Address and Tag Example

Now that we've reviewed the primary ReaderAdapter classes, let's dive into another example by using the reader package to retrieve associated Tags from our IP Address Indicators.

```
1
2 private static void doGetAssociatedTags(Connection conn) throws IOException, FailedResponseException {
3     AbstractIndicatorReaderAdapter reader = ReaderAdapterFactory.createAddressIndicatorReader(conn);
4     List<Address> data = reader.getAll();
5     for (Address address : data) {
6         System.out.printf("IP Address: %20s", address.getIp() );
7
8         List<Tag> associatedTags = reader.getAssociatedTags( address.getIp() );
9         System.out.printf("\tAssociated Tag:");
10        for (Tag tag : associatedTags) {
11            System.out.printf("%20s", tag.getName() );
12        }
13        System.out.println();
14    }
15 }
16
```

Line 3-4 We create an IndicatorReaderAdapter to read all the Addresses the API user has access to. The “getAll()” method returns a collection of Addresses from the ThreatConnect API.

Line 5-6 We iterate through each Address and print out its uniqueId. As mentioned in the section on AbstractIndicatorReaderAdapter's, all uniqueId's for Indicators are Strings. In the case of Address objects, it's the IP Address or the “getIp()” getter method.

Line 8 To get a collection of associated tags for the IP Address, we call the “getAssociatedTags()” method.

Line 10-11 We iterate through each tag returned from the ThreatConnect API for that specific IP Address and print it out to the console.

In this example we learned how to:

- Get a collection of Indicators the API user has access to.
- Retrieve associated data (in this case tags) based on the uniqueId of the Indicator.

## Reader Example 2

TODO



In this example we learned how to:

## Writer Package

The Writer package shares many of the concepts of the Reader package with the distinction of introducing the new functionality of version 2.0 of the ThreatConnect API.

Note the WriterAdapterFactory class is effectively the “hub” for writer adapters. It provides convenience objects for all the adapters in the writer package. Below is a list of the static methods and return types of the **WriterAdapterFactory**.

Class	Description
WriterAdapterFactory	Primary entry point to instantiate all writers in the writer package
AbstractGroupWriterAdapter<T extends Group>	Generic Group Writer Abstract class. Concrete object available in WriterAdapterFactory.
AbstractIndicatorWriterAdapter<T extends Indicator>	Generic Indicator Writer Abstract class. Concrete object available in WriterAdapterFactory.
AbstractWriterAdapter	Base Abstract Writer for all Reader Adapters in the reader package.
SecurityLabelWriterAdapter	Concrete Writer for SecurityLabel data. Convenience object available in WriterAdapterFactory.
TagWriterAdapter	Concrete Writer for Tag data. Convenience object available in WriterAdapterFactory.
VictimWriterAdapter	Concrete Writer for Victim data. Convenience object available in WriterAdapterFactory.

## Writer Factory

The primary methods for the WriterFactory are listed below. They encompass all write functionality for the ThreatConnect API.

Modifier and Type	Method
static AbstractGroupWriterAdapter<Adversary>	createAdversaryGroupWriter(Connection conn)
static AbstractIndicatorWriterAdapter<Address>	createAddressIndicatorWriter(Connection conn)
static AbstractIndicatorWriterAdapter<EmailAddress>	createEmailAddressIndicatorWriter(Connection conn)
static AbstractGroupWriterAdapter<Email>	createEmailGroupWriter(Connection conn)
static AbstractIndicatorWriterAdapter<File>	createFileIndicatorWriter(Connection conn)
static AbstractIndicatorWriterAdapter<Host>	createHostIndicatorWriter(Connection conn)
static AbstractGroupWriterAdapter<Incident>	createIncidentGroupWriter(Connection conn)
static SecurityLabelWriterAdapter	createSecurityLabelWriter(Connection conn)
static AbstractGroupWriterAdapter<Signature>	createSignatureGroupWriter(Connection conn)
static TagWriterAdapter	createTagWriter(Connection conn)
static AbstractGroupWriterAdapter<Threat>	createThreatGroupWriter(Connection conn)
static AbstractIndicatorWriterAdapter<Url>	createUrlIndicatorWriter(Connection conn)
static VictimWriterAdapter	createVictimWriter(Connection conn)

## Writer Responses

Now that we've introduced the WriterFactory, we'll go over some conventions used in the writer API that will help clarify how deletes, creates, and updates are handled by the Java API and what the developer should expect when a failure occurs.

When a single item is modified (create/delete/update) using the Java API, the return type is an ApiEntitySingleResponse object. In an effort to simplify write operation response handling, the ApiEntitySingleResponse object provides a single object for the developer to validate the modify operation.

When a collection of items is modified (create/delete/update) using the Java API, the return type is a WriteListResponse object. Likewise, in an effort to simplify write operation response handling, the WriteListResponse

object holds collections of failed/succeeded ApiEntitySingleResponse objects. The following listing describes how modify responses should be handled.

Modifier and Type	Method	Description
List<ApiEntitySingleResponse>	getFailureList()	Collection of <b>failed</b> ApiEntitySingleResponse objects for each element the API user attempted a write operation to the ThreatConnect API.
List<ApiEntitySingleResponse>	getSuccessList()	Collection of <b>successful</b> ApiEntitySingleResponse objects for each element the API user attempted a write operation to the ThreatConnect API.

The ApiEntitySingleResponse class contains the following relevant methods.

Modifier and Type	Method	Description
boolean	isSuccess()	Returns whether the attempted operation returned successfully from the ThreatConnect API for the item that is part of this response. This should be the first element
String	getMessage()	If "isSuccess()" returns false, check this field to find the cause of the failure for the item that is part of this response.
T	getItem()	This field is a convenience method that returns the item that is part of this response. Note that not all responses return an item.



While the ApiEntitySingleResponse class manages failed write operations to the ThreatConnect API, the developer is responsible for capturing any runtime exceptions that may occur because of network, configuration, or data input issues.

## Writer Create Example

Before we get into the writer package and how to use it to write data to the ThreatConnect API, let's run through a simple example.

```
...
3 import com.cyber2.api.lib.client.writer.AbstractGroupWriterAdapter;
4 import com.cyber2.api.lib.client.writer.WriterAdapterFactory;
5 import com.cyber2.api.lib.conn.Connection;
6 import com.cyber2.api.lib.exception.FailedResponseException;
7 import com.cyber2.api.lib.server.entity.Adversary;
8 import com.cyber2.api.lib.server.response.entity.ApiEntitySingleResponse;
9 import java.io.IOException;
10 import java.util.List;
...
103 private static void doCreate(Connection conn) {
104     AbstractGroupWriterAdapter<Adversary> writer = WriterAdapterFactory.createAdversaryGroupWriter(conn);
105
106     Adversary adversary = new Adversary();
107     adversary.setName("Test Adversary");
108     adversary.setOwnerName("System");
109
110     try {
111         ApiEntitySingleResponse<Adversary,?> response = writer.create(adversary);
112         if ( response.isSuccess() ) {
113             Adversary savedAdversary = response.getItem();
114             System.out.println("Saved: " + savedAdversary.toString() );
115         } else {
116             System.err.println("Error: " + response.getMessage() );
117         }
118     }
119
120     } catch (IOException | FailedResponseException ex) {
121         System.err.println("Error: " + ex.toString());
122     }
123
124 }
```

Line 104        Creates an AbstractGroupWriterAdapter for the Adversary Group type. With this adapter, we can write/update/delete group data elements, victim assets, attributes, and associations.

Line 106-108    Here we create a simple Adversary with a name and owner (organization).

Line 111        We use the writer to create an adversary using the ThreatConnect API. For single item writes, we always get back an ApiEntitySingleResponse object. This object allows us to inspect the response and handle appropriately.

Line 112-114    We call "isSuccess()" to see if the create was successful. If the check passes, we'll get the item associated with the response using the "getItem()" method (Line 113). The successfully saved Adversary object returns from the ThreatConnect API with a valid ID value.

Line 116        If the response is unsuccessful, we print out the response message to the console.

Line 121        We catch any potential runtime exceptions and handle appropriately. In the case of this basic example, we simply dump it out to the console.

In this example we learned how to:

- Create an Adapter using the WriterFactory.
- Create an Adversary, and verify if the save was successful.
- Handle errors from a write operation to the ThreatConnect API.

## Writer Class Overview

Most of the conventions in the reader package are mirrored in the writer package. Much like the reader package, the method naming conventions will be familiar and self-explanatory. [Parameter naming conventions](#) have been kept abstract to allow for a better representation of the identifiers being passed. Below is a listing of the classes in the writer package.

### AbstractGroupWriterAdapter Class

The following methods write data for the Group type (T) linked to this Adapter.

- The create methods require a Group type object as a collection or single object
- The delete methods require the key ID value as a collection or single object
- The update methods require a Group type object as a collection or single object

Modifier and Type	Method and Description
WriteListResponse<T>	create(List<T> itemList)
ApiEntitySingleResponse	create(T item)
ApiEntitySingleResponse	create(T item, String ownerName)
WriteListResponse<P>	delete(List<P> itemIds)
WriteListResponse<P>	delete(List<P> itemIds, String ownerName)
ApiEntitySingleResponse	delete(P itemId)
ApiEntitySingleResponse	delete(P itemId, String ownerName)
WriteListResponse<T>	update(List<T> itemList)

WriteListResponse<T>	update(List<T> itemList, String ownerName)
ApiEntitySingleResponse	update(T item)
ApiEntitySingleResponse	update(T item, String ownerName)

## Associate Groups

The methods below associate a Group type to another Group type. We associate groups by passing in the uniqueId (Integer) with the group id that we want to associate to.

Modifier and Type	Method
WriteListResponse<Integer>	associateGroupAdversaries(Integer uniqueId, List<Integer> adversaryIds)
WriteListResponse<Integer>	associateGroupAdversaries(Integer uniqueId, List<Integer> adversaryIds, String ownerName)
ApiEntitySingleResponse	associateGroupAdversary(Integer uniqueId, Integer adversaryId)
ApiEntitySingleResponse	associateGroupAdversary(Integer uniqueId, Integer adversaryId, String ownerName)
ApiEntitySingleResponse	associateGroupEmail(Integer uniqueId, Integer emailId)
ApiEntitySingleResponse	associateGroupEmail(Integer uniqueId, Integer emailId, String ownerName)
WriteListResponse<Integer>	associateGroupEmails(Integer uniqueId, List<Integer> emailIds)
WriteListResponse<Integer>	associateGroupEmails(Integer uniqueId, List<Integer> emailIds, String ownerName)
ApiEntitySingleResponse	associateGroupIncident(Integer uniqueId, Integer incidentId)
ApiEntitySingleResponse	associateGroupIncident(Integer uniqueId, Integer incidentId, String ownerName)
WriteListResponse<Integer>	associateGroupIncidents(Integer uniqueId, List<Integer> incidentIds)
WriteListResponse<Integer>	associateGroupIncidents(Integer uniqueId, List<Integer> incidentIds, String ownerName)
ApiEntitySingleResponse	associateGroupSignature(Integer uniqueId, Integer signatureId)
ApiEntitySingleResponse	associateGroupSignature(Integer uniqueId, Integer signatureId, String ownerName)

WriteListResponse<Integer>	associateGroupSignatures(Integer uniqueId, List<Integer> signatureIds)
WriteListResponse<Integer>	associateGroupSignatures(Integer uniqueId, List<Integer> signatureIds, String ownerName)
ApiEntitySingleResponse	associateGroupThreat(Integer uniqueId, Integer threatId)
ApiEntitySingleResponse	associateGroupThreat(Integer uniqueId, Integer threatId, String ownerName)
WriteListResponse<Integer>	associateGroupThreats(Integer uniqueId, List<Integer> threatIds)
WriteListResponse<Integer>	associateGroupThreats(Integer uniqueId, List<Integer> threatIds, String ownerName)

### *Associate Indicators*

The following methods associate Indicators to the Group type.

Modifier and Type	Method
ApiEntitySingleResponse	associateIndicatorAddress(Integer uniqueId, String ipAddress)
ApiEntitySingleResponse	associateIndicatorAddress(Integer uniqueId, String ipAddress, String ownerName)
WriteListResponse<String>	associateIndicatorAddresses(Integer uniqueId, List<String> ipAddresses)
WriteListResponse<String>	associateIndicatorAddresses(Integer uniqueId, List<String> ipAddresses, String ownerName)
ApiEntitySingleResponse	associateIndicatorEmailAddress(Integer uniqueId, String emailAddress)
ApiEntitySingleResponse	associateIndicatorEmailAddress(Integer uniqueId, String emailAddress, String ownerName)
WriteListResponse<String>	associateIndicatorEmailAddresses(Integer uniqueId, List<String> emailAddresses)
WriteListResponse<String>	associateIndicatorEmailAddresses(Integer uniqueId, List<String> emailAddresses, String ownerName)
ApiEntitySingleResponse	associateIndicatorFile(Integer uniqueId, String fileHash)
ApiEntitySingleResponse	associateIndicatorFile(Integer uniqueId, String fileHash, String ownerName)
WriteListResponse<String>	associateIndicatorFiles(Integer uniqueId, List<String> fileHashes)

WriteListResponse<String>	associateIndicatorFiles(Integer uniqueId, List<String> fileHashes, String ownerName)
ApiEntitySingleResponse	associateIndicatorHost(Integer uniqueId, String hostName)
ApiEntitySingleResponse	associateIndicatorHost(Integer uniqueId, String hostName, String ownerName)
WriteListResponse<String>	associateIndicatorHosts(Integer uniqueId, List<String> hostNames)
WriteListResponse<String>	associateIndicatorHosts(Integer uniqueId, List<String> hostNames, String ownerName)
ApiEntitySingleResponse	associateIndicatorUrl(Integer uniqueId, String urlText)
ApiEntitySingleResponse	associateIndicatorUrl(Integer uniqueId, String urlText, String ownerName)
WriteListResponse<String>	associateIndicatorUrls(Integer uniqueId, List<String> urlTexts)
WriteListResponse<String>	associateIndicatorUrls(Integer uniqueId, List<String> urlTexts, String ownerName)

## *Associate Security Labels*

The following methods associate security labels to the Group type.

Modifier and Type	Method
ApiEntitySingleResponse	associateSecurityLabel(Integer uniqueId, String securityLabel)
ApiEntitySingleResponse	associateSecurityLabel(Integer uniqueId, String securityLabel, String ownerName)
WriteListResponse<String>	associateSecurityLabels(Integer uniqueId, List<String> securityLabels)
WriteListResponse<String>	associateSecurityLabels(Integer uniqueId, List<String> securityLabels, String ownerName)

## *Associate Tag*

The following methods associate tags to the Group type.



Modifier and Type	Method
ApiEntitySingleResponse	associateTag(Integer uniqueId, String tagName)
ApiEntitySingleResponse	associateTag(Integer uniqueId, String tagName, String ownerName)
WriteListResponse<String>	associateTags(Integer uniqueId, List<String> tagNames)
WriteListResponse<String>	associateTags(Integer uniqueId, List<String> tagNames, String ownerName)

### *Associate Victim*

The following methods associate Victims to the Group type.

Modifier and Type	Method
ApiEntitySingleResponse	associateVictim(Integer uniqueId, Integer victimId)
ApiEntitySingleResponse	associateVictim(Integer uniqueId, Integer victimId, String ownerName)
WriteListResponse<Integer>	associateVictims(Integer uniqueId, List<Integer> victimIds)
WriteListResponse<Integer>	associateVictims(Integer uniqueId, List<Integer> victimIds, String ownerName)

### *Associate Victim Asset*

The following methods associate VictimAssets to the Group type.

Modifier and Type	Method
ApiEntitySingleResponse	associateVictimAssetEmailAddress(Integer uniqueId, Integer assetId)
ApiEntitySingleResponse	associateVictimAssetEmailAddress(Integer uniqueId, Integer assetId, String ownerName)
WriteListResponse<Integer>	associateVictimAssetEmailAddresses(Integer uniqueId, List<Integer> assetIds)
WriteListResponse<Integer>	associateVictimAssetEmailAddresses(Integer uniqueId, List<Integer> assetIds, String ownerName)
ApiEntitySingleResponse	associateVictimAssetNetworkAccount(Integer uniqueId, Integer assetId)

ApiEntitySingleResponse	associateVictimAssetNetworkAccount(Integer uniqueId, Integer assetId, String ownerName)
WriteListResponse<Integer>	associateVictimAssetNetworkAccounts(Integer uniqueId, List<Integer> assetIds)
WriteListResponse<Integer>	associateVictimAssetNetworkAccounts(Integer uniqueId, List<Integer> assetIds, String ownerName)
ApiEntitySingleResponse	associateVictimAssetPhoneNumber(Integer uniqueId, Integer assetId)
ApiEntitySingleResponse	associateVictimAssetPhoneNumber(Integer uniqueId, Integer assetId, String ownerName)
WriteListResponse<Integer>	associateVictimAssetPhoneNumbers(Integer uniqueId, List<Integer> assetIds)
WriteListResponse<Integer>	associateVictimAssetPhoneNumbers(Integer uniqueId, List<Integer> assetIds, String ownerName)
ApiEntitySingleResponse	associateVictimAssetSocialNetwork(Integer uniqueId, Integer assetId)
ApiEntitySingleResponse	associateVictimAssetSocialNetwork(Integer uniqueId, Integer assetId, String ownerName)
WriteListResponse<Integer>	associateVictimAssetSocialNetworks(Integer uniqueId, List<Integer> assetIds)
WriteListResponse<Integer>	associateVictimAssetSocialNetworks(Integer uniqueId, List<Integer> assetIds, String ownerName)
ApiEntitySingleResponse	associateVictimAssetWebsite(Integer uniqueId, Integer assetId)
ApiEntitySingleResponse	associateVictimAssetWebsite(Integer uniqueId, Integer assetId, String ownerName)
WriteListResponse<Integer>	associateVictimAssetWebsites(Integer uniqueId, List<Integer> assetIds)
WriteListResponse<Integer>	associateVictimAssetWebsites(Integer uniqueId, List<Integer> assetIds, String ownerName)

## Add Attributes

The following set of methods add Attribute types to a Group.

Modifier and Type	Method
ApiEntitySingleResponse	addAttribute(Integer uniqueId, Attribute attribute)
ApiEntitySingleResponse	addAttribute(Integer uniqueId, Attribute attribute, String ownerName)
WriteListResponse<Attribute>	addAttributes(Integer uniqueId, List<Attribute> attributes)

WriteListResponse<Attribute>	addAttributes(Integer uniqueId, List<Attribute> attribute, String ownerName)
ApiEntitySingleResponse	addAttributeSecurityLabel(Integer uniqueId, Integer attributeId, String securityLabel)
ApiEntitySingleResponse	addAttributeSecurityLabel(Integer uniqueId, Integer attributeId, String securityLabel, String ownerName)
WriteListResponse<String>	addAttributeSecurityLabels(Integer uniqueId, Integer attributeId, List<String> securityLabels)
WriteListResponse<String>	addAttributeSecurityLabels(Integer uniqueId, Integer attributeId, List<String> securityLabels, String ownerName)

### *Update Attribute*

The following methods update an Attribute added to a specific Group type.

Modifier and Type	Method
ApiEntitySingleResponse	updateAttribute(Integer uniqueId, Attribute attribute)
ApiEntitySingleResponse	updateAttribute(Integer uniqueId, Attribute attribute, String ownerName)
WriteListResponse<Attribute>	updateAttributes(Integer uniqueId, List<Attribute> attributes)
WriteListResponse<Attribute>	updateAttributes(Integer uniqueId, List<Attribute> attribute, String ownerName)

### *Delete Group Association*

The following methods **delete** Group associations to a specific Group type.

Modifier and Type	Method
WriteListResponse<Integer>	deleteAssociatedGroupAdversaries(Integer uniqueId, List<Integer> adversaryIds)

WriteListResponse<Integer>	deleteAssociatedGroupAdversaries(Integer uniqueId, List<Integer> adversaryIds, String ownerName)
ApiEntitySingleResponse	deleteAssociatedGroupAdversary(Integer uniqueId, Integer adversaryId)
ApiEntitySingleResponse	deleteAssociatedGroupAdversary(Integer uniqueId, Integer adversaryId, String ownerName)
ApiEntitySingleResponse	deleteAssociatedGroupEmail(Integer uniqueId, Integer emailId)
ApiEntitySingleResponse	deleteAssociatedGroupEmail(Integer uniqueId, Integer emailId, String ownerName)
WriteListResponse<Integer>	deleteAssociatedGroupEmails(Integer uniqueId, List<Integer> emailIds)
WriteListResponse<Integer>	deleteAssociatedGroupEmails(Integer uniqueId, List<Integer> emailIds, String ownerName)
ApiEntitySingleResponse	deleteAssociatedGroupIncident(Integer uniqueId, Integer incidentId)
ApiEntitySingleResponse	deleteAssociatedGroupIncident(Integer uniqueId, Integer incidentId, String ownerName)
WriteListResponse<Integer>	deleteAssociatedGroupIncidents(Integer uniqueId, List<Integer> incidentIds)
WriteListResponse<Integer>	deleteAssociatedGroupIncidents(Integer uniqueId, List<Integer> incidentIds, String ownerName)
ApiEntitySingleResponse	deleteAssociatedGroupSignature(Integer uniqueId, Integer signatureId)
ApiEntitySingleResponse	deleteAssociatedGroupSignature(Integer uniqueId, Integer signatureId, String ownerName)
WriteListResponse<Integer>	deleteAssociatedGroupSignatures(Integer uniqueId, List<Integer> signatureIds)
WriteListResponse<Integer>	deleteAssociatedGroupSignatures(Integer uniqueId, List<Integer> signatureIds, String ownerName)
ApiEntitySingleResponse	deleteAssociatedGroupThreat(Integer uniqueId, Integer threatId)
ApiEntitySingleResponse	deleteAssociatedGroupThreat(Integer uniqueId, Integer threatId, String ownerName)
WriteListResponse<Integer>	deleteAssociatedGroupThreats(Integer uniqueId, List<Integer> threatIds)
WriteListResponse<Integer>	deleteAssociatedGroupThreats(Integer uniqueId, List<Integer> threatIds, String ownerName)

## Delete Indicator Associations

The following methods will delete Indicator associations to a specific Group type.

Modifier and Type	Method
ApiEntitySingleResponse	deleteAssociatedIndicatorAddress(Integer uniqueId, String ipAddress)
ApiEntitySingleResponse	deleteAssociatedIndicatorAddress(Integer uniqueId, String ipAddress, String ownerName)
WriteListResponse<String>	deleteAssociatedIndicatorAddresses(Integer uniqueId, List<String> ipAddresses)
WriteListResponse<String>	deleteAssociatedIndicatorAddresses(Integer uniqueId, List<String> ipAddresses, String ownerName)
ApiEntitySingleResponse	deleteAssociatedIndicatorEmailAddress(Integer uniqueId, String emailAddress)
ApiEntitySingleResponse	deleteAssociatedIndicatorEmailAddress(Integer uniqueId, String emailAddress, String ownerName)
WriteListResponse<String>	deleteAssociatedIndicatorEmailAddresses(Integer uniqueId, List<String> emailAddresses)
WriteListResponse<String>	deleteAssociatedIndicatorEmailAddresses(Integer uniqueId, List<String> emailAddresses, String ownerName)
ApiEntitySingleResponse	deleteAssociatedIndicatorFile(Integer uniqueId, String fileHash)
ApiEntitySingleResponse	deleteAssociatedIndicatorFile(Integer uniqueId, String fileHash, String ownerName)
WriteListResponse<String>	deleteAssociatedIndicatorFiles(Integer uniqueId, List<String> fileHashes)
WriteListResponse<String>	deleteAssociatedIndicatorFiles(Integer uniqueId, List<String> fileHashes, String ownerName)
ApiEntitySingleResponse	deleteAssociatedIndicatorHost(Integer uniqueId, String hostName)
ApiEntitySingleResponse	deleteAssociatedIndicatorHost(Integer uniqueId, String hostName, String ownerName)
WriteListResponse<String>	deleteAssociatedIndicatorHosts(Integer uniqueId, List<String> hostNames)

WriteListResponse<String>	deleteAssociatedIndicatorHosts(Integer uniqueId, List<String> hostNames, String ownerName)
ApiEntitySingleResponse	deleteAssociatedIndicatorUrl(Integer uniqueId, String urlText)
ApiEntitySingleResponse	deleteAssociatedIndicatorUrl(Integer uniqueId, String urlText, String ownerName)
WriteListResponse<String>	deleteAssociatedIndicatorUrls(Integer uniqueId, List<String> urlTexts)
WriteListResponse<String>	deleteAssociatedIndicatorUrls(Integer uniqueId, List<String> urlTexts, String ownerName)

### *Delete Security Label Associations*

The following methods will **delete** SecurityLabel associations to a specific Group type.

Modifier and Type	Method
WriteListResponse<String>	deleteAssociatedSecurityLabel(Integer uniqueId, List<String> securityLabels)
WriteListResponse<String>	deleteAssociatedSecurityLabel(Integer uniqueId, List<String> securityLabels, String ownerName)
ApiEntitySingleResponse	deleteAssociatedSecurityLabel(Integer uniqueId, String securityLabel)
ApiEntitySingleResponse	deleteAssociatedSecurityLabel(Integer uniqueId, String securityLabel, String ownerName)

### *Delete Tag Associations*

The following methods will **delete** Tag associations to a specific Group type.

Modifier and Type	Method
ApiEntitySingleResponse	deleteAssociatedTag(Integer uniqueId, String tagName)

ApiEntitySingleResponse	deleteAssociatedTag(Integer uniqueId, String tagName, String ownerName)
WriteListResponse<String>	deleteAssociatedTags(Integer uniqueId, List<String> tagNames)
WriteListResponse<String>	deleteAssociatedTags(Integer uniqueId, List<String> tagNames, String ownerName)

### *Delete Victim Associations*

The following methods will **delete** Victim associations to a specific Group type.

Modifier and Type	Method
ApiEntitySingleResponse	deleteAssociatedVictim(Integer uniqueId, Integer victimId)
ApiEntitySingleResponse	deleteAssociatedVictim(Integer uniqueId, Integer victimId, String ownerName)
WriteListResponse<Integer>	deleteAssociatedVictims(Integer uniqueId, List<Integer> victimIds)
WriteListResponse<Integer>	deleteAssociatedVictims(Integer uniqueId, List<Integer> victimIds, String ownerName)

### *Delete VictimAsset Associations*

The following methods will **delete** VictimAsset associations to a specific Group type.

Modifier and Type	Method
ApiEntitySingleResponse	deleteAssociatedVictimAssetEmailAddress(Integer uniqueId, Integer assetId)
ApiEntitySingleResponse	deleteAssociatedVictimAssetEmailAddress(Integer uniqueId, Integer assetId, String ownerName)
WriteListResponse<Integer>	deleteAssociatedVictimAssetEmailAddresses(Integer uniqueId, List<Integer> assetIds)
WriteListResponse<Integer>	deleteAssociatedVictimAssetEmailAddresses(Integer uniqueId, List<Integer> assetIds, String ownerName)
ApiEntitySingleResponse	deleteAssociatedVictimAssetNetworkAccount(Integer uniqueId, Integer assetId)
ApiEntitySingleResponse	deleteAssociatedVictimAssetNetworkAccount(Integer uniqueId, Integer assetId, String ownerName)

WriteListResponse<Integer>	deleteAssociatedVictimAssetNetworkAccounts(Integer uniqueId, List<Integer> assetIds)
WriteListResponse<Integer>	deleteAssociatedVictimAssetNetworkAccounts(Integer uniqueId, List<Integer> assetIds, String ownerName)
ApiEntitySingleResponse	deleteAssociatedVictimAssetPhoneNumber(Integer uniqueId, Integer assetId)
ApiEntitySingleResponse	deleteAssociatedVictimAssetPhoneNumber(Integer uniqueId, Integer assetId, String ownerName)
WriteListResponse<Integer>	deleteAssociatedVictimAssetPhoneNumbers(Integer uniqueId, List<Integer> assetIds)
WriteListResponse<Integer>	deleteAssociatedVictimAssetPhoneNumbers(Integer uniqueId, List<Integer> assetIds, String ownerName)
ApiEntitySingleResponse	deleteAssociatedVictimAssetSocialNetwork(Integer uniqueId, Integer assetId)
ApiEntitySingleResponse	deleteAssociatedVictimAssetSocialNetwork(Integer uniqueId, Integer assetId, String ownerName)
WriteListResponse<Integer>	deleteAssociatedVictimAssetSocialNetworks(Integer uniqueId, List<Integer> assetIds)
WriteListResponse<Integer>	deleteAssociatedVictimAssetSocialNetworks(Integer uniqueId, List<Integer> assetIds, String ownerName)
ApiEntitySingleResponse	deleteAssociatedVictimAssetWebsite(Integer uniqueId, Integer assetId)
ApiEntitySingleResponse	deleteAssociatedVictimAssetWebsite(Integer uniqueId, Integer assetId, String ownerName)
WriteListResponse<Integer>	deleteAssociatedVictimAssetWebsites(Integer uniqueId, List<Integer> assetIds)
WriteListResponse<Integer>	deleteAssociatedVictimAssetWebsites(Integer uniqueId, List<Integer> assetIds, String ownerName)

## Delete Attribute

The following methods **delete** attributes from a specific Group type.

Modifier and Type	Method
ApiEntitySingleResponse	deleteAttribute(Integer uniqueId, Integer attribute)
ApiEntitySingleResponse	deleteAttribute(Integer uniqueId, Integer attribute, String ownerName)
WriteListResponse<Integer>	deleteAttributes(Integer uniqueId, List<Integer> attributes)



WriteListResponse<Integer>	deleteAttributes(Integer uniqueId, List<Integer> attribute, String ownerName)
ApiEntitySingleResponse	deleteAttributeSecurityLabel(Integer uniqueId, Integer attributeId, String securityLabel)
ApiEntitySingleResponse	deleteAttributeSecurityLabel(Integer uniqueId, Integer attributeId, String securityLabel, String ownerName)
WriteListResponse<String>	deleteAttributeSecurityLabels(Integer uniqueId, Integer attributeId, List<String> securityLabels)
WriteListResponse<String>	deleteAttributeSecurityLabels(Integer uniqueId, Integer attributeId, List<String> securityLabels, String ownerName)

## Update Attribute

The following methods **update** the attribute of the specific Group type.

Modifier and Type	Method
ApiEntitySingleResponse	updateAttribute(Integer uniqueId, Attribute attribute)
ApiEntitySingleResponse	updateAttribute(Integer uniqueId, Attribute attribute, String ownerName)
WriteListResponse<Attribute>	updateAttributes(Integer uniqueId, List<Attribute> attributes)
WriteListResponse<Attribute>	updateAttributes(Integer uniqueId, List<Attribute> attribute, String ownerName)

## AbstractIndicatorWriterAdapter

The AbstractIndicatorWriterAdapter share most of the write functionality with the AbstractGroupWriterAdapter. In fact, they both implement the following writer interfaces.

Interface
AttributeAssociateWritable<T>
GroupAssociateWritable<T>

IndicatorAssociateWritable<T>
SecurityLabelAssociateWritable<T>
TagAssociateWritable<T>
VictimAssetAssociateWritable<T>

These interfaces allow the AbstractIndicatorWriterAdapter to run all of the same methods as the AbstractGroupWriterAdapter see [that section](#) for more detail.



The key parameter-level distinction between the AbstractIndicatorWriterAdapter and the AbstractGroupWriterAdapter is the type (T) for the uniqueId parameter. As mentioned in previous sections, Indicator uniqueId types are all String's. The method naming conventions are the same.

There is one additional subclass of the AbstractIndicatorWriterAdapter that has a concrete implementation. This class is the **FileIndicatorWriterAdapter**. It obviously has all the functionality of the AbstractIndicatorWriterAdapter with the addition of the following write methods.

Modifier and Type	Method
FileOccurrence	updateFileOccurrence(String fileHash, FileOccurrence fileOccurrence)
FileOccurrence	updateFileOccurrence(String fileHash, FileOccurrence fileOccurrence, String ownerName)
WriteListResponse<FileOccurrence>	updateFileOccurrences(String fileHash, List<FileOccurrence> fileOccurrences)
WriteListResponse<FileOccurrence>	updateFileOccurrences(String fileHash, List<FileOccurrence> fileOccurrences, String ownerName)

### *SecurityLabelWriterAdapter*

The SecurityLabelWriterAdapter class allows [Group](#) and [Indicator](#) associations. Much like the Indicator Adapters, the uniqueId is a user-created security label String. In addition to creating associations, the SecurityLabelWriterAdapter allows deleting associations from [Group](#) and [Indicator](#) types..

Below is the standard create methods available to all WriterAdapter's. Note that the deletes require the Security Label as the "uniqueId" String (P). The create and update requires the full SecurityLabel object (T).

Modifier and Type	Method
WriteListResponse<T>	create(List<T> itemList)
ApiEntitySingleResponse	create(T item)
ApiEntitySingleResponse	create(T item, String ownerName)
WriteListResponse<P>	delete(List<P> itemIds)
WriteListResponse<P>	delete(List<P> itemIds, String ownerName)
ApiEntitySingleResponse	delete(P itemId)
ApiEntitySingleResponse	delete(P itemId, String ownerName)
WriteListResponse<T>	update(List<T> itemList)
WriteListResponse<T>	update(List<T> itemList, String ownerName)
ApiEntitySingleResponse	update(T item)
ApiEntitySingleResponse	update(T item, String ownerName)

### *TagWriterAdapter*

The TagWriterAdapter class allows [Group](#) and [Indicator](#) associations. Much like the Indicator Adapters, the uniqueId is a user-created tag name String. In addition to creating associations, the TagWriterAdapter allows deleting associations from [Group](#) and [Indicator](#) types.

Below is the standard create methods available to all WriterAdapter's. Note that the deletes require the Tag Name as the "uniqueId" String (P). The create and update requires the full Tag object (T).

Modifier and Type	Method
WriteListResponse<T>	create(List<T> itemList)

ApiEntitySingleResponse	create(T item)
ApiEntitySingleResponse	create(T item, String ownerName)
WriteListResponse<P>	delete(List<P> itemIds)
WriteListResponse<P>	delete(List<P> itemIds, String ownerName)
ApiEntitySingleResponse	delete(P itemId)
ApiEntitySingleResponse	delete(P itemId, String ownerName)
WriteListResponse<T>	update(List<T> itemList)
WriteListResponse<T>	update(List<T> itemList, String ownerName)
ApiEntitySingleResponse	update(T item)
ApiEntitySingleResponse	update(T item, String ownerName)

### *VictimWriterAdapter*

The TagWriterAdapter class allows [Group](#), [Indicator](#), and [VictimAsset](#) associations. Much like the Group Adapters, the uniqueId is a user-created security label String. In addition to creating associations, the VictimAssetWriterAdapter can remove associations for [Group](#), [Indicator](#), and [VictimAssets](#).

Below is the standard create methods available to all WriterAdapter's. Note that the deletes require the system generated VictimAsset ID as the "uniqueId" Integer (P). The create and update requires the full VictimAsset object (T).

Modifier and Type	Method
WriteListResponse<T>	create(List<T> itemList)
ApiEntitySingleResponse	create(T item)
ApiEntitySingleResponse	create(T item, String ownerName)
WriteListResponse<P>	delete(List<P> itemIds)
WriteListResponse<P>	delete(List<P> itemIds, String ownerName)

ApiEntitySingleResponse	delete(P itemId)
ApiEntitySingleResponse	delete(P itemId, String ownerName)
WriteListResponse<T>	update(List<T> itemList)
WriteListResponse<T>	update(List<T> itemList, String ownerName)
ApiEntitySingleResponse	update(T item)
ApiEntitySingleResponse	update(T item, String ownerName)

## Writer Examples

Now that we've covered the writer package, let's go over creating, deleting, and updating data using the ThreatConnect Java API.

### Writer Delete Example

```

...
2
3 import com.cyber2.api.lib.client.reader.AbstractGroupReaderAdapter;
4 import com.cyber2.api.lib.client.reader.ReaderAdapterFactory;
5
6 import com.cyber2.api.lib.client.writer.AbstractGroupWriterAdapter;
7 import com.cyber2.api.lib.client.writer.WriterAdapterFactory;
8 import com.cyber2.api.lib.server.response.entity.ApiEntitySingleResponse;
9
10 import com.cyber2.api.lib.server.entity.Adversary;
11 import com.cyber2.api.lib.conn.Connection;
12 import java.io.IOException;
13
...
130 private static void doDelete(Connection conn) {
131     AbstractGroupWriterAdapter<Adversary> writer = WriterAdapterFactory.createAdversaryGroupWriter(conn);
132
133     Adversary adversary = new Adversary();
134     adversary.setName("Test Adversary");
135     adversary.setOwnerName("System");
136
137     try {
138         ApiEntitySingleResponse<Adversary,?> createResponse = writer.create(adversary);
139         if ( createResponse.isSuccess() ) {
140             System.out.println("Saved: " + createResponse.getItem() );
141             ApiEntitySingleResponse<Adversary,?> deleteResponse
142                 = writer.delete( createResponse.getItem().getId() );
143             if ( deleteResponse.isSuccess() ) {
144                 System.out.println("Deleted: " + createResponse.getItem() );

```

```

145         } else {
146             System.err.println("Delete Failed. Cause: " + deleteResponse.getMessage() );
147         }
148     } else {
149         System.err.println("Create Failed. Cause: " + createResponse.getMessage());
150     }
151 }
152 } catch (IOException | FailedResponseException ex) {
153     System.err.println("Error: " + ex.toString());
154 }
155 }
156 ...

```

- Line 131 Since we'll be creating and deleting Adversary objects from the ThreatConnect API, let's instantiate an AbstractGroupWriterAdapter with the Adversary parameterized type applied.
- Line 138-139 Here we create an Adversary object with the ThreatConnect API. We capture the response and check if the save was successful by calling "isSuccess()".
- Line 140 We print out the response Adversary object returned from the ThreatConnect API. The "getItem()" method will return this object with the ID field populated. This method will always hold the saved item on a successful response.
- Line 141-142 We use the ID from the successful create to delete the same Adversary object. Note the call to the "delete()" method requires the system generated Adversary ID.
- Line 143-144 We check if the delete response was successful and dump the original response.
- Line 146 With a failed delete, we can print out the error message by calling the "getMessage()" method on the response object.
- Line 149 If the original create failed, we do the same as a failed delete and call the "getMessage()" method to find the cause.

## Writer Update Example

```

...
2
3 import com.cyber2.api.lib.client.reader.AbstractGroupReaderAdapter;
4 import com.cyber2.api.lib.client.reader.ReaderAdapterFactory;
5 import com.cyber2.api.lib.client.writer.AbstractGroupWriterAdapter;
6 import com.cyber2.api.lib.client.writer.WriterAdapterFactory;
7 import com.cyber2.api.lib.conn.Connection;
8 import com.cyber2.api.lib.exception.FailedResponseException;
9 import com.cyber2.api.lib.server.entity.Adversary;
...
15 import com.cyber2.api.lib.server.response.entity.ApiEntitySingleResponse;
16 import java.io.IOException;
17 import java.util.List;

```

```

18
...
153
154     private static void doUpdate(Connection conn) {
155         AbstractGroupWriterAdapter<Adversary> writer = WriterAdapterFactory.createAdversaryGroupWriter(conn);
156
157         Adversary adversary = new Adversary();
158         adversary.setName("Test Adversary");
159         adversary.setOwnerName("System");
160
161         try {
162             ApiEntitySingleResponse<Adversary,?> createResponse = writer.create(adversary);
163             if ( createResponse.isSuccess() ) {
164                 System.out.println("Created Adversary: " + createResponse.getItem() );
165
166                 Adversary updatedAdversary = createResponse.getItem();
167                 updatedAdversary.setName("UPDATED: " + createResponse.getItem().getName() );
168                 System.out.println("Saving Updated Adversary: " + updatedAdversary );
169
170                 ApiEntitySingleResponse<Adversary,?> updateResponse = writer.update( updatedAdversary );
171                 if ( updateResponse.isSuccess() ) {
172                     System.out.println("Updated Adversary: " + updateResponse.getItem() );
173                 } else {
174                     System.err.println("Failed to Update Adversary: " + updateResponse.getMessage() );
175                 }
176             } else {
177                 System.err.println("Failed to Create Adversary: " + createResponse.getMessage() );
178             }
179         } catch (IOException | FailedResponseException ex) {
180             System.err.println("Error: " + ex.toString());
181         }
182     }
183
184 }
..

```

Lines 155-164 As we did with the delete example, we create a test Adversary and save it to the ThreatConnect API.

Lines 166-168 We assign the created Adversary to a variable called “updatedAdversary()” so we can change the name of the Adversary (line 167). Before we update the Adversary in ThreatConnect, we print it out to the console. The output should have an ID value populated and the name should read: “UPDATED: Test Adversary”.

Lines 170-172 At this point we call the “update()” method to save the changes to ThreatConnect. The argument to this method is the actual Adversary object. Just like the delete, we check the response for success and write it to the console.

## Writer Add Attribute Example

```

1
2     private static Email createTestEmail() {
3         Email email = new Email();
4         email.setName("Test Email");
5         email.setOwnerName("System");

```

```

6      email.setFrom("admin@test.com");
7      email.setTo("test@test.com");
8      email.setSubject("Test Subject");
9      email.setBody("Test Body");
10     email.setHeader("Test Header");
11
12     return email;
13 }
14
15 private static Attribute createTestAttribute() {
16     Attribute attribute = new Attribute();
17     attribute.setSource("Test Source");
18     attribute.setDisplayed(true);
19     attribute.setType("Description");
20     attribute.setValue("Test Attribute Description");
21
22     return attribute;
23 }
...
69 private static void doAddAttribute(Connection conn) {
70     AbstractGroupWriterAdapter<Email> writer = WriterAdapterFactory.createEmailGroupWriter(conn);
71
72     Email email = createTestEmail();
73     Attribute attribute = createTestAttribute();
74
75     try {
76         // -----
77         // Create Email
78         // -----
79         ApiEntitySingleResponse<Email, ?> createResponse = writer.create(email);
80         if (createResponse.isSuccess()) {
81             System.out.println("Created Email: " + createResponse.getItem());
82
83             // -----
84             // Add Attribute
85             // -----
86             ApiEntitySingleResponse<Attribute, ?> attribResponse
87                 = writer.addAttribute( createResponse.getItem().getId(), attribute );
88
89             if ( attribResponse.isSuccess() ) {
90                 System.out.println("\tAdded Attribute: " + attribResponse.getItem() );
91             } else {
92                 System.err.println("Failed to Add Attribute: " + attribResponse.getMessage());
93             }
94         } else {
95             System.err.println("Failed to Create Email: " + createResponse.getMessage());
96         }
97     }
98
99     } catch (IOException | FailedResponseException ex) {
100         System.err.println("Error: " + ex.toString());
101     }
102
103 }
104

```

Lines 72-73      Create test email object and the attribute we will add.

Line 79-81      Here we create the email in ThreatConnect and check if it was successful.

Line 86-87      The “addAttribute()” method takes the email group id and the attribute object we want to add.



Line 89-90      To confirm we added the attribute successfully, we check the response.

## Writer Associate Indicator Example

```
1
2  private static Email createTestEmail() {
3      Email email = new Email();
4      email.setName("Test Email");
5      email.setOwnerName("System");
6      email.setFrom("admin@test.com");
7      email.setTo("test@test.com");
8      email.setSubject("Test Subject");
9      email.setBody("Test Body");
10     email.setHeader("Test Header");
11
12     return email;
13 }
14
...
25  private static Host createTestHost() {
26      Host host = new Host();
27      host.setOwnerName("System");
28      host.setDescription("Test Host");
29      host.setHostName("www.bad-hostname.com");
30      host.setRating( 5.0 );
31      host.setConfidence(98.0);
32
33     return host;
34 }
35
...
104
105  private static void doAssociateIndicator(Connection conn) {
106      AbstractGroupWriterAdapter<Email> gWriter= WriterAdapterFactory.createEmailGroupWriter(conn);
107      AbstractIndicatorWriterAdapter<Host> hWriter = WriterAdapterFactory.createHostIndicatorWriter(conn);
108
109      Email email = createTestEmail();
110      Host host = createTestHost();
111
112      try {
113
114          // -----
115          // Create Email and Host
116          // -----
117          ApiEntitySingleResponse<Email, ?> createResponseEmail = gWriter.create(email);
118          ApiEntitySingleResponse<Host, ?> createResponseHost = hWriter.create(host);
119          if (createResponseEmail.isSuccess() && createResponseHost.isSuccess() ) {
120              System.out.println("Created Email: " + createResponseEmail.getItem());
121              System.out.println("Created Host: " + createResponseHost.getItem());
122
123              // -----
124              // Associate Host
125              // -----
126              ApiEntitySingleResponse assocResponse
127                  = gWriter.associateIndicatorHost(createResponseEmail.getItem().getId()
128                                                  , createResponseHost.getItem().getHostName() );
129
130              if ( assocResponse.isSuccess() ) {
131                  System.out.println("\tAssociated Host: " + createResponseHost.getItem().getHostName() );
132              } else {
133                  System.err.println("Failed to Add Attribute: " + assocResponse.getMessage());
134              }
135          }
```

```

136         } else {
137             if ( !createResponseEmail.isSuccess() ) {
138                 System.err.println("Failed to Create Email: " + createResponseEmail.getMessage());
139             }
140             if ( !createResponseHost.isSuccess() ) {
141                 System.err.println("Failed to Create Host: " + createResponseHost.getMessage());
142             }
143         }
144     }
145     } catch (IOException | FailedResponseException ex) {
146         System.err.println("Error: " + ex.toString());
147     }
148 }
149 }

```

- Line 106-107 We create two writers -- one for the email we want to create and the other is for the host we want to associate.
- Line 109-110 Create test email object and the host we will associate.
- Line 117-121 To set up our example, we create both email and host in ThreatConnect. We also check to ensure the create was successful and print the items to the console.
- Line 126-128 We associate the host to the email group by using the email ID and the hostname (the unique ID for the host indicator).
- Line 130-133 Here we verify the association was successful.

## Writer Associate Group Example

```

...
2     private static Email createTestEmail() {
3         Email email = new Email();
4         email.setName("Test Email");
5         email.setOwnerName("System");
6         email.setFrom("admin@test.com");
7         email.setTo("test@test.com");
8         email.setSubject("Test Subject");
9         email.setBody("Test Body");
10        email.setHeader("Test Header");
11    }
12    return email;
13 }
14
...
36    private static Threat createTestThreat() {
37        Threat threat = new Threat();
38        threat.setOwnerName("System");
39        threat.setName("Test Threat");
40    }
41    return threat;
42 }
...
151
152    private static void doAssociateGroup(Connection conn) {

```

```

153 AbstractGroupWriterAdapter<Email> gWriter= WriterAdapterFactory.createEmailGroupWriter(conn);
154 AbstractGroupWriterAdapter<Threat> tWriter = WriterAdapterFactory.createThreatGroupWriter(conn);
155
156 Email email = createTestEmail();
157 Threat threat = createTestThreat();
158
159 try {
160     // -----
161     // Create Email and Threat
162     // -----
163     ApiEntitySingleResponse<Email, ?> createResponseEmail = gWriter.create(email);
164     ApiEntitySingleResponse<Threat, ?> createResponseThreat = tWriter.create(threat);
165     if (createResponseEmail.isSuccess() && createResponseThreat.isSuccess() ) {
166         System.out.println("Created Email: " + createResponseEmail.getItem());
167         System.out.println("Created Threat: " + createResponseThreat.getItem());
168
169         // -----
170         // Associate Threat
171         // -----
172         ApiEntitySingleResponse assocResponse
173             = gWriter.associateGroupThreat(createResponseEmail.getItem().getId(),
174                                           createResponseThreat.getItem().getId());
175
176         if ( assocResponse.isSuccess() ) {
177             System.out.println("\tAssociated Threat: " + createResponseThreat.getItem().getId() );
178         } else {
179             System.err.println("Failed to Associate Threat: " + assocResponse.getMessage());
180         }
181     } else {
182         if ( !createResponseEmail.isSuccess() ) {
183             System.err.println("Failed to Create Email: " + createResponseEmail.getMessage());
184         }
185         if ( !createResponseThreat.isSuccess() ) {
186             System.err.println("Failed to Create Threat: " + createResponseThreat.getMessage());
187         }
188     }
189 }
190
191 } catch (IOException | FailedResponseException ex) {
192     System.err.println("Error: " + ex.toString());
193 }
194
195 }
196

```

Line 153-154 We create two writers -- one for the email we want to create and the other is for the threat we want to associate.

Line 156-157 Create test email object and the threat we will associate.

Line 163-167 To set up our example, we create both email and threat in ThreatConnect. We also check to ensure the create was successful and print the items to the console.

Line 172-174 We associate the threat to the email group by using the email ID and the threat ID.

Line 176-179 Here we verify the association was successful.

## Writer Associate Tag Example

```
...
 2  private static Email createTestEmail() {
 3      Email email = new Email();
 4      email.setName("Test Email");
 5      email.setOwnerName("System");
 6      email.setFrom("admin@test.com");
 7      email.setTo("test@test.com");
 8      email.setSubject("Test Subject");
 9      email.setBody("Test Body");
10      email.setHeader("Test Header");
11
12      return email;
13  }
14
...
44  private static Tag createTestTag() {
45      Tag tag = new Tag();
46      tag.setName("Test-Tag");
47      tag.setDescription("Test Tag Description");
48
49      return tag;
50  }
51
...
196
197  private static void doAssociateTag(Connection conn) {
198      AbstractGroupWriterAdapter<Email> gWriter= WriterAdapterFactory.createEmailGroupWriter(conn);
199      TagWriterAdapter tWriter = WriterAdapterFactory.createTagWriter(conn);
200
201      Email email = createTestEmail();
202      Tag tag = createTestTag();
203
204      try {
205          // -----
206          // Create Email and Tag
207          // -----
208          ApiEntitySingleResponse<Email, ?> createResponseEmail = gWriter.create(email);
209          tWriter.delete(tag.getName()); // delete if it exists
210          ApiEntitySingleResponse<Tag, ?> createResponseTag = tWriter.create(tag);
211
212          if (createResponseEmail.isSuccess() && createResponseTag.isSuccess() ) {
213              System.out.println("Created Email: " + createResponseEmail.getItem());
214              System.out.println("Created Tag: " + createResponseTag.getItem());
215
216              // -----
217              // Associate Tag
218              // -----
219              ApiEntitySingleResponse assocResponse
220                  = gWriter.associateTag(createResponseEmail.getItem().getId()
221                                          , createResponseTag.getItem().getName() );
222
223              if ( assocResponse.isSuccess() ) {
224                  System.out.println("\tAssociated Tag: " + createResponseTag.getItem().getName() );
225              } else {
226                  System.err.println("Failed to Associate Tag: " + assocResponse.getMessage());
227              }
228
229          } else {
230              if ( !createResponseEmail.isSuccess() ) {
231                  System.err.println("Failed to Create Email: " + createResponseEmail.getMessage());
232              }
233              if ( !createResponseTag.isSuccess() ) {
234                  System.err.println("Failed to Create Tag: " + createResponseTag.getMessage());
235              }
236          }
237      }
238  }
239
...
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
271
```

```

236     }
237
238     } catch (IOException | FailedResponseException ex) {
239         System.err.println("Error: " + ex.toString());
240     }
241 }

```

- Line 198-199 We create two writers -- one for the email we want to create and the other is for the tag we want to associate.
- Line 201-202 Create test email object and the tag we will associate.
- Line 208-214 To set up our example, we create both email and tag in ThreatConnect. We also check to ensure the create was successful and print the items to the console.
- Line 219-221 We associate the tag to the email group by using the email ID and the tag name (the unique ID for the tag).
- Line 223-226 Here we verify the association was successful.

## Writer Associate Victim Example

```

...
59
60     private static Victim createTestVictim() {
61         Victim victim = new Victim();
62         victim.setOrg("System");
63         victim.setName("Test API Victim");
64         victim.setDescription("Test API Victim Description");
65
66         return victim;
67     }
...
304     private static void doAssociateVictim(Connection conn) {
305         AbstractGroupWriterAdapter<Email> gWriter= WriterAdapterFactory.createEmailGroupWriter(conn);
306         VictimWriterAdapter vWriter = WriterAdapterFactory.createVictimWriter(conn);
307
308         Email email = createTestEmail();
309         Victim victim = createTestVictim();
310
311         try {
312             // -----
313             // Create Email and Victim
314             // -----
315             ApiEntitySingleResponse<Email, ?> createResponseEmail = gWriter.create(email);
316             ApiEntitySingleResponse<Victim, ?> createResponseVictim = vWriter.create(victim);
317             if (createResponseEmail.isSuccess() && createResponseVictim.isSuccess() ) {
318                 System.out.println("Created Email: " + createResponseEmail.getItem());
319                 System.out.println("Created Victim: " + createResponseVictim.getItem());
320
321                 // -----
322                 // Associate Victim
323                 // -----
324                 ApiEntitySingleResponse assocResponse
325                     = gWriter.associateVictim(createResponseEmail.getItem().getId()

```

```

326         , createResponseVictim.getItem().getId());
327
328         if ( assocResponse.isSuccess() ) {
329             System.out.println("\tAssociated Victim: " + createResponseVictim.getItem().getId() );
330         } else {
331             System.err.println("Failed to Associate Victim: " + assocResponse.getMessage());
332         }
333
334     } else {
335         if ( !createResponseEmail.isSuccess() ) {
336             System.err.println("Failed to Create Email: " + createResponseEmail.getMessage());
337         }
338         if ( !createResponseVictim.isSuccess() ) {
339             System.err.println("Failed to Create Victim: " + createResponseVictim.getMessage());
340         }
341     }
342
343 } catch (IOException | FailedResponseException ex) {
344     System.err.println("Error: " + ex.toString());
345 }
346
347 }

```

- Line 305-306 We create two writers -- one for the email we want to create and the other is for the victim we want to associate.
- Line 308-309 Create test email object and the victim we will associate.
- Line 315-319 To set up our example, we create both email and victim in ThreatConnect. We also check to ensure the create was successful and print the items to the console.
- Line 324-326 We associate the victim to the email group by using the email ID and the victim ID.
- Line 328-331 Here we verify the association was successful.

## Writer Remove Association Example

```

...
243 private static void doRemoveAssociatedTag(Connection conn) {
244
245     AbstractGroupWriterAdapter<Email> gWriter= WriterAdapterFactory.createEmailGroupWriter(conn);
246     TagWriterAdapter tWriter = WriterAdapterFactory.createTagWriter(conn);
247
248     Email email = createTestEmail();
249     Tag tag = createTestTag();
250
251     try {
252         // -----
253         // Create Email and Tag
254         // -----
255         ApiEntitySingleResponse<Email, ?> createResponseEmail = gWriter.create(email);
256         tWriter.delete(tag.getName()); // delete if it exists
257         ApiEntitySingleResponse<Tag, ?> createResponseTag = tWriter.create(tag);
258

```

```

259         if (createResponseEmail.isSuccess() && createResponseTag.isSuccess() ) {
260             System.out.println("Created Email: " + createResponseEmail.getItem());
261             System.out.println("Created Tag: " + createResponseTag.getItem());
262
263             // -----
264             // Associate Tag
265             // -----
266             ApiEntitySingleResponse assocResponse
267                 = gWriter.associateTag(createResponseEmail.getItem().getId()
268                                     , createResponseTag.getItem().getName() );
269
270             if ( assocResponse.isSuccess() ) {
271                 System.out.println("\tAssociated Tag: " + createResponseTag.getItem().getName() );
272
273                 // -----
274                 // Delete Association
275                 // -----
276                 ApiEntitySingleResponse deleteAssocResponse
277                     = gWriter.deleteAssociatedTag(createResponseEmail.getItem().getId()
278                                                 , createResponseTag.getItem().getName() );
279
280                 if ( deleteAssocResponse.isSuccess() ) {
281                     System.out.println("\tDeleted Associated Tag: "
↪ + createResponseTag.getItem().getName() );
282                 } else {
283                     System.err.println("Failed to delete Associated Tag: "
↪ + deleteAssocResponse.getMessage());
284                 }
285
286             } else {
287                 System.err.println("Failed to Associate Tag: " + assocResponse.getMessage());
288             }
289
290         } else {
291             if ( !createResponseEmail.isSuccess() ) {
292                 System.err.println("Failed to Create Email: "
↪ + createResponseEmail.getMessage());
293             }
294             if ( !createResponseTag.isSuccess() ) {
295                 System.err.println("Failed to Create Tag: " + createResponseTag.getMessage());
296             }
297         }
298
299     } catch (IOException | FailedResponseException ex) {
300         System.err.println("Error: " + ex.toString());
301     }
302 }

```

Lines 245-271 As we have reviewed in prior examples, we create and associate the email group with the tag item.

Line 276-278 Here we call the “deleteAssociatedTag()” method using the same parameters as the associate method. We use the email item ID value with the tag name.

Line 280-281 Finally we check that the deleting was successful and dump out the message to the console.

In the previous two examples we learned how to:

→ Delete and update an Adversary and verify it was saved successfully to ThreatConnect.

- Add an Attribute to a Group item.
- Associate Indicators, Groups, Tags, and Victims.
- Remove Association from a Group item.