

THOMSON REUTERS CALAISMODEL ABSTRACTION LAYER

DEVELOPER GUIDE



© Thomson Reuters 2017. All Rights Reserved.

Thomson Reuters, by publishing this document, does not guarantee that any information contained herein is and will remain accurate or that use of the information will ensure correct and faultless operation of the relevant service or equipment. Thomson Reuters, its agents and employees, shall not be held liable to or through any user for any loss or damage whatsoever resulting from reliance on the information contained herein.

This document contains information proprietary to Thomson Reuters and may not be reproduced, disclosed, or used in whole or part without the express written permission of Thomson Reuters.

Any Software, including but not limited to, the code, screen, structure, sequence, and organization thereof, and Documentation are protected by national copyright laws and international treaty provisions. This manual is subject to U.S. and other national export regulations.

Nothing in this document is intended, nor does it, alter the legal obligations, responsibilities or relationship between yourself and Thomson Reuters as set out in the contract existing between us.

Contents

Chapter 1	Introduction	1
Chapter 2	Using the Abstraction Layer	2
2.1	Setting up the Environment (Dependencies)	3
2.2	CalaisModelCreator	4
2.3	CalaisModel	5
2.4	CalaisObject	10
Chapter 3	Example Code	22
3.1	Example 1: Retrieving a List of Metadata Types	23
3.2	Example 2: Retrieving all the Objects of a Specified Type	25
3.3	Example 3: Filtering the Object List and Retrieving Specified Related Data	26

Chapter 1 Introduction

The CalaisModel Abstraction Layer provides a client API that maps RDF/XML entities to Java objects. The Abstraction Layer includes the CalaisModelCreator, a tool used to automatically generate an instance of the CalaisModel, populated with your data. This instance of the CalaisModel enables access to the RDF data without writing any logic to process XML elements and without any specific knowledge of the underlying data structure.

The CalaisModel Abstraction Layer may be used with any new or legacy Open Calais, Intelligent Tagging, or One Calais version, package, or profile.

The Abstraction Layer includes the following libraries:

- Java Library
- .NET Library
- Python Library

Download the Abstraction Layer Package from the API page of www.opencalais.com.

Refer to the Thomson Reuters Open Calais API User Guide, also available on the API page of www.opencalais.com, for detailed information about the metadata found in the RDF output.

Note: If you are using the Java Library, please see [Setting up the Environment \(Dependencies\)](#).

Chapter 2 Using the Abstraction Layer

The Abstraction Layer package includes the following three main classes:

[CalaisModelCreator](#)

[CalaisModel](#)

[CalaisObject](#)

2.1 Setting up the Environment (Dependencies)

The CalaisModel uses the following third party libraries to parse the RDF: Jena, slf4j, commons-lang.

You can work with Apache Maven or add the CalaisModel.jar and the required dependencies to your project.

To work with Apache Maven:

1. Download the Abstraction Layer from the API page of www.opencalais.com.
2. From the downloaded Abstraction Layer package, extract **clf-calaisModel.java.jar**, and then upload it to your artifactory, using the details illustrated in the next step.
3. Add the dependency to your pom.xml file.

```
<dependency>
  <groupId> com.tr.tms.abstractionLayer</groupId>
  <artifactId>calaisModel.java</artifactId>
  <version>The release number of the Abstraction Layer. (See the Note, below.)</version>
</dependency>
```

4. Add the dependency of slf4j implementation. For example: logback.

Note: The release number of the Abstraction Layer is listed in the pom.xml file, in the <version> tag. Find the pom.xml file in this location: OneCalais Abstraction Library Java/oncalais/clf-calaisModel.java.jar/META-INF/maven/com.tr.tms.abstractionLayer/calaisModel.java/pom.xml

2.2 CalaisModelCreator

The CalaisModelCreator is used to generate an instance of the CalaisModel, populated with your data.

2.2.1 Initializing the CalaisModel

Initialize the CalaisModel by using the utility class **CalaisModelCreator**.

CalaisModelCreator contains three utility methods that should be used to create the CalaisModel:

- `readFile` – used when the RDF is a file.
- `readText` – used when the RDF is a string.
- `readInputStream` – used when the RDF comes from an input stream.

Example:

```
CalaisModel calaisModel = CalaisModelCreator.readFile("C:\\input\\myRdf.xml", Format.RDF);
```

Note: An *IllegalArgumentException* is thrown if the CalaisModel fails to load. In this case, it is likely that the RDF is faulty. Check your RDF.

2.3 CalaisModel

The CalaisModel contains all the CalaisObjects, and enables retrieving CalaisObjects by ID and by Type. Note that CalaisObjects correspond to the rdf:Description nodes.

2.3.1 Methods

[getCalaisObjectById](#)

[getCalaisObjectByType](#)

[getAllTypes](#)

[getAllCalaisObjects](#)

2.3.1.1 getCalaisObjectById

Purpose	Used to retrieve a specified CalaisObject.
Syntax	<code>calaisModel.getCalaisObjectById("objectId");</code>
Input Parameters	Name: objectId Type: string Description: Unique identifying URL obtained from the <rdf:about> attribute of the object (node) in the RDF file.
Return Value	The CalaisObject with the specified ID.
Examples	<pre>CalaisObject calaisObject = calaisModel.getCalaisObjectById("http://d.opencalais.com/er/company/ralg- oa/4295861160");</pre> <pre>CalaisObject resolvedCompany = calaisModel.getCalaisObjectById("http://d.opencalais.com/er/company/ralg- oa/4295861160");</pre>
Remarks	Use the entire object ID as it appears in the <rdf:about> attribute. For example, http://d.opencalais.com/er/company/ralg-oa/4295861160

2.3.1.2 getCalaisObjectByType

Purpose	Used to retrieve all objects of a specified type.
Syntax	<code>calaisModel.getCalaisObjectByType("type");</code>
Input Parameters	Name: type Type: string Description: Identifying URI from the <rdf:type> attribute of the object (node) in the RDF file, or from the getAllTypes list.
Return Value	A list of CalaisObjects of the specified type.
Examples	<pre>CalaisObject calaisObject = calaisModel.getCalaisObjectByType("http://s.opencalais.com/1/type/em/e/Company");</pre> <pre>CalaisObject resolvedCompany = calaisModel.getCalaisObjectByType("http://s.opencalais.com/1/type/er/Company");</pre>
Remarks	Use the entire rdf:resource as it appears in the <rdf:type> attribute. For example, "http://s.opencalais.com/1/type/em/e/Company"

2.3.1.3 getAllTypes

Purpose	Used to retrieve a list of all of the types found in the RDF.
Syntax	<code>List<String>Types =calaisModel.getAllTypes();</code>
Input Parameters	None
Return Value	A list containing all the <rdf:type> attribute values found in the RDF.
Example	<pre>List<String>Types = calaisModel.getAllTypes();</pre>

2.3.1.4 getAllCalaisObjects

Purpose	Used to retrieve all of the CalaisObjects. (The objects correspond to the RDF nodes.)
Syntax	List<CalaisObject> calaisObjects = calaisModel.getAllCalaisObjects();
Input Parameters	None
Return Value	A list containing all the CalaisObjects.
Example	<pre>List<CalaisObject> calaisObjects = calaisModel.getAllCalaisObjects();</pre>

2.4 CalaisObject

[CalaisObject Infrastructure](#)

[CalaisObject Methods](#)

2.4.1 Infrastructure

Each CalaisObject in the model corresponds to an RDF node.

Every CalaisObject contains the following:

- Objectid – A unique ID taken from the *rdf:about* attribute of the relevant RDF node.
- Type – Identifies the object type, as defined by the *rdf:type* attribute of the relevant RDF node.
- Map Attributes (key-values) – The CalaisObject contains the following 4 maps, described in the following sections:
 - [Literals](#)
 - [References](#)
 - [ExternalURIs](#)
 - [BackReferences](#)

2.4.1.1 Literals

Description	A map that contains all the RDF node attributes that have a string value.										
Syntax	Map<String, List<String>>literals										
Map Key	The attribute name.										
Map Value	A list of attribute values (the list may contain one or more values, according to whether the attribute appears one or more times in the RDF node). A calculated value, e.g. a confidence score is also treated as a Literals.										
Example 1	<pre> <rdf: Description rdf: about="http://d.opencalais.com/generichasher-1/6d615c9f-563e-354f-90b4-9306c13bf521"> <rdf: type rdf: resource="http://s.opencalais.com/1/type/em/r/Conviction"/> <c: forenduserdisplay>false</c: forenduserdisplay> <!-- Ted Stevens --> <c: person rdf: resource="http://d.opencalais.com/pershash-1/aef43f3d-e3d8-3795-a5e1-5fea1fb1fcd0"/> <c: charge>violating federal ethics laws</c: charge> <c: charge>corruption</c: charge> <c: date>2009-03-16</c: date> <c: datestring>on Monday</c: datestring> </rdf: Description> </pre> <p>Literals Map:</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value (List of Strings)</th> </tr> </thead> <tbody> <tr> <td>http://s.opencalais.com/1/pred/forenduserdisplay</td> <td>[false]</td> </tr> <tr> <td>http://s.opencalais.com/1/pred/charge</td> <td>[violating federal ethics laws, corruption]</td> </tr> <tr> <td>http://s.opencalais.com/1/pred/date</td> <td>[2009-03-16]</td> </tr> <tr> <td>http://s.opencalais.com/1/pred/datestring</td> <td>[on Monday]</td> </tr> </tbody> </table>	Key	Value (List of Strings)	http://s.opencalais.com/1/pred/forenduserdisplay	[false]	http://s.opencalais.com/1/pred/charge	[violating federal ethics laws, corruption]	http://s.opencalais.com/1/pred/date	[2009-03-16]	http://s.opencalais.com/1/pred/datestring	[on Monday]
Key	Value (List of Strings)										
http://s.opencalais.com/1/pred/forenduserdisplay	[false]										
http://s.opencalais.com/1/pred/charge	[violating federal ethics laws, corruption]										
http://s.opencalais.com/1/pred/date	[2009-03-16]										
http://s.opencalais.com/1/pred/datestring	[on Monday]										

<p>Example 2</p>	<pre><rdf: Description rdf: about="http://d.opencalais.com/er/company/ralg-0a/4295905573"> <rdf: type rdf: resource="http://s.opencalais.com/1/type/er/Company" /> <c: docId rdf: resource="http://d.opencalais.com/dochash-1/7586b818-40af-3d55-ac16-bf3520cddfa6" /> <c: permid>4295905573</c: permid> <c: score>0.9928677</c: score> <!-- Apple --> <c: subject rdf: resource="http://d.opencalais.com/comphash-1/705cd5cf-93e1-323c-8d4e-1ea3200d37e4" /> <c: legacyid rdf: resource="http://d.opencalais.com/er/company/ralg-tr1r/23d07771-c50b-315b-8050-3cdaf47ac0d0" /> <c: name>Apple Inc</c: name> <c: commonname>Apple</c: commonname> <c: ticker>AAPL</c: ticker> <c: primaryric>AAPL.OQ</c: primaryric> <c: openpermid rdf: resource="https://permid.org/1-4295905573" /> </rdf: Description></pre> <p>Literals Map:</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value (List of Strings)</th> </tr> </thead> <tbody> <tr> <td>http://s.opencalais.com/1/pred/permid</td> <td>[4295905573]</td> </tr> <tr> <td>http://s.opencalais.com/1/pred/score</td> <td>[0.9928677]</td> </tr> <tr> <td>http://s.opencalais.com/1/pred/name</td> <td>[Apple Inc]</td> </tr> <tr> <td>http://s.opencalais.com/1/pred/commonname</td> <td>[Apple]</td> </tr> <tr> <td>http://s.opencalais.com/1/pred/ticker</td> <td>[AAPL]</td> </tr> <tr> <td>http://s.opencalais.com/1/pred/primaryric</td> <td>[AAPL.OQ]</td> </tr> </tbody> </table>	Key	Value (List of Strings)	http://s.opencalais.com/1/pred/permid	[4295905573]	http://s.opencalais.com/1/pred/score	[0.9928677]	http://s.opencalais.com/1/pred/name	[Apple Inc]	http://s.opencalais.com/1/pred/commonname	[Apple]	http://s.opencalais.com/1/pred/ticker	[AAPL]	http://s.opencalais.com/1/pred/primaryric	[AAPL.OQ]
Key	Value (List of Strings)														
http://s.opencalais.com/1/pred/permid	[4295905573]														
http://s.opencalais.com/1/pred/score	[0.9928677]														
http://s.opencalais.com/1/pred/name	[Apple Inc]														
http://s.opencalais.com/1/pred/commonname	[Apple]														
http://s.opencalais.com/1/pred/ticker	[AAPL]														
http://s.opencalais.com/1/pred/primaryric	[AAPL.OQ]														
<p>Remarks</p>	<p>The attribute name must be the full name, e.g. <i>http://s.opencalais.com/1/pred/name</i>, and not just "name," or "c:name."</p>														

2.4.1.2 References

Description	A URI resource that refers to another object (node) within the RDF file. (An internal reference.)								
Syntax	Map<String, List<CalaisObject>>references								
Map Key	The attribute name.								
Map Value	A list of CalaisObject references.								
Example 1	<pre> <rdf: Description rdf: about="http://d.opencalais.com/er/company/ralg-0a/4295905573"> <rdf: type rdf: resource="http://s.opencalais.com/1/type/er/Company" /> <c: docId rdf: resource="http://d.opencalais.com/dochash-1/7586b818-40af-3d55-ac16-bf3520cddfa6" /> <c: permId>4295905573</c: permId> <c: score>0.9928677</c: score> <!-- Apple --> <c: subject rdf: resource="http://d.opencalais.com/comphash-1/705cd5cf-93e1-323c-8d4e-1ea3200d37e4" /> <c: legacyId rdf: resource="http://d.opencalais.com/er/company/ralg-tr1r/23d07771-c50b-315b-8050-3cdaf47ac0d0" /> <c: name>Apple Inc</c: name> <c: commonname>Apple</c: commonname> <c: ticker>AAPL</c: ticker> <c: primaryric>AAPL.0Q</c: primaryric> <c: openpermId rdf: resource="https://permId.org/1-4295905573" /> </rdf: Description> </pre> <p>References Map</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value (List of References to CalaisObjects)</th> </tr> </thead> <tbody> <tr> <td>http://s.opencalais.com/1/pred/docId</td> <td>[http://d.opencalais.com/dochash-1/7586b818-40af-3d55-ac16-bf3520cddfa6]</td> </tr> <tr> <td>http://s.opencalais.com/1/pred/subject</td> <td>[http://d.opencalais.com/comphash-1/705cd5cf-93e1-323c-8d4e-1ea3200d37e4]</td> </tr> <tr> <td>http://s.opencalais.com/1/pred/legacyid</td> <td>[http://d.opencalais.com/er/company/ralg-tr1r/23d07771-c50b-315b-8050-3cdaf47ac0d0]</td> </tr> </tbody> </table>	Key	Value (List of References to CalaisObjects)	http://s.opencalais.com/1/pred/docId	[http://d.opencalais.com/dochash-1/7586b818-40af-3d55-ac16-bf3520cddfa6]	http://s.opencalais.com/1/pred/subject	[http://d.opencalais.com/comphash-1/705cd5cf-93e1-323c-8d4e-1ea3200d37e4]	http://s.opencalais.com/1/pred/legacyid	[http://d.opencalais.com/er/company/ralg-tr1r/23d07771-c50b-315b-8050-3cdaf47ac0d0]
Key	Value (List of References to CalaisObjects)								
http://s.opencalais.com/1/pred/docId	[http://d.opencalais.com/dochash-1/7586b818-40af-3d55-ac16-bf3520cddfa6]								
http://s.opencalais.com/1/pred/subject	[http://d.opencalais.com/comphash-1/705cd5cf-93e1-323c-8d4e-1ea3200d37e4]								
http://s.opencalais.com/1/pred/legacyid	[http://d.opencalais.com/er/company/ralg-tr1r/23d07771-c50b-315b-8050-3cdaf47ac0d0]								

<p>Example 2</p>	<pre><rdf: Description rdf: about="http://d.opencalais.com/generichasher-1/8d357a11-aa13-3c49-8345-f0dabe9bfefd"> <rdf: type rdf: resource="http://s.opencalais.com/1/type/em/r/Alliance"/> <c: forenduserdisplay>true</c: forenduserdisplay> <!-- Wells Fargo & Company --> <c: company rdf: resource="http://d.opencalais.com/comphash-1/dd1abc5b-4804-3398-9db8-c193b8c2cc57"/> <!-- Metavante Corporation --> <c: company rdf: resource="http://d.opencalais.com/comphash-1/172bcdf6-783c-325a-a2aa-87bc6c2ec3e3"/> <c: date>2009-03-19</c: date> <c: datestring>today</c: datestring> <c: status>announced</c: status> </rdf: Description></pre> <p>References Map</p> <table border="1"> <thead> <tr> <th data-bbox="344 730 922 772">Key</th> <th data-bbox="922 730 1498 772">Value (List of References to CalaisObjects)</th> </tr> </thead> <tbody> <tr> <td data-bbox="344 772 922 892">http://s.opencalais.com/1/pred/company</td> <td data-bbox="922 772 1498 892">[http://d.opencalais.com/comphash-1/dd1abc5b-4804-3398-9db8-c193b8c2cc57, http://d.opencalais.com/comphash-1/172bcdf6-783c-325a-a2aa-87bc6c2ec3e3]</td> </tr> </tbody> </table>	Key	Value (List of References to CalaisObjects)	http://s.opencalais.com/1/pred/company	[http://d.opencalais.com/comphash-1/dd1abc5b-4804-3398-9db8-c193b8c2cc57, http://d.opencalais.com/comphash-1/172bcdf6-783c-325a-a2aa-87bc6c2ec3e3]
Key	Value (List of References to CalaisObjects)				
http://s.opencalais.com/1/pred/company	[http://d.opencalais.com/comphash-1/dd1abc5b-4804-3398-9db8-c193b8c2cc57, http://d.opencalais.com/comphash-1/172bcdf6-783c-325a-a2aa-87bc6c2ec3e3]				
<p>Remarks</p>	<p>The attribute name must be the full name, e.g. <i>http://s.opencalais.com/1/pred/company</i>, and not just “<i>comopany</i>,” or “<i>c:company</i>.”</p>				

2.4.1.3 ExternalURIs

Description	A URI resource that points to something external to the RDF file (anything on the Internet).				
Syntax	Map<String, List<String>>externalURIs				
Map Key	The attribute name.				
Map Value	A list of URI values.				
Example	<pre><rdf: Description rdf: about="http://d.opencalais.com/er/company/ralg-0a/4295905573"> <rdf: type rdf: resource="http://s.opencalais.com/1/type/er/Company" /> <c: docId rdf: resource="http://d.opencalais.com/dochash-1/7586b818-40af-3d55-ac16-bf3520cddfa6" /> <c: permid>4295905573</c: permid> <c: score>0.9928677</c: score> <!-- Apple --> <c: subject rdf: resource="http://d.opencalais.com/comphash-1/705cd5cf-93e1-323c-8d4e-1ea3200d37e4" /> <c: legacyid rdf: resource="http://d.opencalais.com/er/company/ralg-tr1r/23d07771-c50b-315b-8050-3cdaf47ac0d0" /> <c: name>Apple Inc</c: name> <c: commonname>Apple</c: commonname> <c: ticker>AAPL</c: ticker> <c: primaryrc>AAPL.0Q</c: primaryrc> <c: openpermid rdf: resource="https://permid.org/1-4295905573" /> </rdf: Description></pre> <p>ExternalURIs Map:</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value (List of URI Strings)</th> </tr> </thead> <tbody> <tr> <td>http://s.opencalais.com/1/pred/openpermid</td> <td>[https://permid.org/1-4295905573]</td> </tr> </tbody> </table>	Key	Value (List of URI Strings)	http://s.opencalais.com/1/pred/openpermid	[https://permid.org/1-4295905573]
Key	Value (List of URI Strings)				
http://s.opencalais.com/1/pred/openpermid	[https://permid.org/1-4295905573]				
Remarks	The attribute name must be the full name, e.g. <i>http://s.opencalais.com/1/pred/openpermid</i> , and not just “ <i>openpermid</i> ,” or “ <i>c:openpermid</i> .”				

2.4.1.4 BackReferences

Description	A URI resource that refers to a node in the RDF file that points to this object (node).
Syntax	Map<String,Map<String, List<CalaisObject>>>backReferences
Map Key	The name of the attribute (in the other node) that points to this node (object).
Map Value	A nested map of key-values: key = the node type (of the other node that points to this node) Value = list of the CalaisObjects that point to this object (node)
Example	<p>The RDF node that corresponds to the CalaisObject:</p> <pre><rdf: Description rdf: about="http://d.opencalais.com/comphash-1/dd1abc5b-4804-3398-9db8-c193b8c2cc57"> <rdf: type rdf: resource="http://s.opencalais.com/1/type/em/e/Company" /> <c: forenduserdisplay>true</c: forenduserdisplay> <c: name>Wells Fargo & Company</c: name> <c: nationality>N/A</c: nationality> <c: confidencelevel>0.977</c: confidencelevel> </rdf: Description></pre> <p>Other nodes in the RDF that point to it:</p> <pre><rdf: Description rdf: about="http://d.opencalais.com/dochash-1/5e2f7843-087b-3535-9baa-35778f5b9e97/Instance/32"> <rdf: type rdf: resource="http://s.opencalais.com/1/type/sys/InstanceInfo" /> <c: docId rdf: resource="http://d.opencalais.com/dochash-1/5e2f7843-087b-3535-9baa-35778f5b9e97" /> <c: subject rdf: resource="http://d.opencalais.com/comphash-1/dd1abc5b-4804-3398-9db8-c193b8c2cc57"/> <!--Company: Wells Fargo & Company; --> <c: detection>[Corporation, First Union Corporation and]Wells Fargo & Company[, today announced a strategic alliance with]</c: detection> <c: prefix>Corporation, First Union Corporation and </c: prefix> <c: exact>Wells Fargo & Company</c: exact> <c: suffix>, today announced a strategic alliance with</c: suffix> <c: offset>487</c: offset> <c: length>25</c: length> </rdf: Description></pre> <pre><rdf: Description rdf: about="http://d.opencalais.com/er/company/ralg-0a/8589934175"> <rdf: type rdf: resource="http://s.opencalais.com/1/type/er/Company" /> <c: docId rdf: resource="http://d.opencalais.com/dochash-1/5e2f7843-087b-3535-9baa-35778f5b9e97" /> <c: score>0.97700715</c: score> <!--Wells Fargo & Company--> <c: subject rdf: resource="http://d.opencalais.com/comphash-1/dd1abc5b-4804-3398-9db8-c193b8c2cc57"/> <c: name>Wells Fargo & Co</c: name> <c: commonname>Wells Far</c: commonname> <c: ticker>WFC</c: ticker> <c: primaryc>WFC.N</c: primaryc> <c: permid rdf: resource="http://10.11.11.15/1-8589934175" /> </rdf: Description></pre>

BackReferences Map:							
Key	Value (Map)						
http://s.opencalais.com/1/pred/subject	<table border="1"> <thead> <tr> <th>Key</th> <th>Value (List of References to CalaisObjects)</th> </tr> </thead> <tbody> <tr> <td>http://s.opencalais.com/1/type/sy/s/instanceInfo</td> <td>[http://d.opencalais.com/dochash-1/5e2f7843-087b-3535-9baa-35778f5b9e97/Instance/32]</td> </tr> <tr> <td>http://s.opencalais.com/1/type/er/Company</td> <td>http://d.opencalais.com/er/company/ralg-0a/8589934175</td> </tr> </tbody> </table>	Key	Value (List of References to CalaisObjects)	http://s.opencalais.com/1/type/sy/s/instanceInfo	[http://d.opencalais.com/dochash-1/5e2f7843-087b-3535-9baa-35778f5b9e97/Instance/32]	http://s.opencalais.com/1/type/er/Company	http://d.opencalais.com/er/company/ralg-0a/8589934175
Key	Value (List of References to CalaisObjects)						
http://s.opencalais.com/1/type/sy/s/instanceInfo	[http://d.opencalais.com/dochash-1/5e2f7843-087b-3535-9baa-35778f5b9e97/Instance/32]						
http://s.opencalais.com/1/type/er/Company	http://d.opencalais.com/er/company/ralg-0a/8589934175						
Remarks	The attribute name must be the full name, e.g. http://s.opencalais.com/1/type/tag/Confidence and not just "Confidence."						

2.4.2 Methods

[getObjectId](#)

[getType](#)

[getLiterals](#)

[getReferences](#)

[getExternalURIs](#)

[getBackReferences](#)

[getBackReferencesByFieldName](#)

2.4.2.1 getObjectld

Purpose	Used to retrieve the object ID.
Syntax	<code>String objectId = calaisObject.getObjectld();</code>
Input Parameters	None
Return Value	The object ID.

2.4.2.2 getType

Purpose	Used to retrieve the object type.
Syntax	<code>String type = calaisObject.getType();</code>
Input Parameters	None
Return Value	The object type.

2.4.2.3 getLiterals

Purpose	Used to retrieve the Literals map.
Syntax	<code>Map<String, List<String>> literals = calaisObject.getLiterals();</code>
Input Parameters	None
Return Value	Literals map

2.4.2.4 getReferences

Purpose	Used to retrieve the References map.
Syntax	<code>Map<String, List<CalaisObject>> references = calaisObject.getReferences();</code>
Input Parameters	None
Return Value	References map.

2.4.2.5 getExternalURIs

Purpose	Used to retrieve ExternalURIs map.
Syntax	<code>Map<String, List<String>>externalURIs = calaisObject.getExternalURIs();</code>
Input Parameters	None
Return Value	ExternalURIs map.

2.4.2.6 getBackReferences

Purpose	Used to retrieve the BackReferences map.
Syntax	<code>Map<String,Map<String, List<CalaisObject>>> backReferences = calaisObject.getBackReferences();</code>
Input Parameters	None
Return Value	BackReferences map.

2.4.2.7 getBackReferencesByFieldName

Purpose	Used to retrieve a subset of the BackReferences map that includes key values for a specified fieldname attribute.
Syntax	<code>Map<String, List<CalaisObject>> getBackReferencesByFieldName(String fieldName);</code>
Input Parameters	Name: fieldName Type: string Description: The relevant field (RDF attribute) name.
Return Value	A subset of the BackReferences map that includes the key-values for the specified fieldname attribute.
Example	<code>Map<String, List<CalaisObject>> subjectBackReferences = calaisObject.getBackReferencesByFieldName("http://s.opencalais.com/1/pred/subject");</code>
Return Value	Subset of the BackReferences map that includes the key-values for the specified fieldname attribute.

Chapter 3 Example Code

When files are sent to TRIT for processing, TRIT analyzes the input text, and identifies and tags mentions of things (entities) like companies, people, deals, and events based on a list of predefined metadata types.

The default tagging output is in RDF format. At first glance the RDF output seems complicated and hard to understand. But once you familiarize yourself with the metadata type structure, the RDF layout becomes easy to work with.

Please see the API User Guide for both a conceptual overview and detailed descriptions of all the metadata types.

Now we are going to walk you through some examples of using the abstraction layer to get data from the RDF output.

We are going to illustrate how to:

- [Retrieve a list of Metadata Types from the RDF output](#)
- [Retrieve all the objects of a specified type](#)
- [Filter the object list and get related data](#)

3.1 Example 1: Retrieving a List of Metadata Types

The following code snippet uses the CalaisModel method “getAllTypes” to read the TRIT RDF output and return a list of all the metadata types it contains:

```
import com.tr.tms.abstractionLayer.CalaisModel;
import com.tr.tms.abstractionLayer.CalaisModelCreator;
import com.tr.tms.abstractionLayer.engine.Engine;

import java.util.Set;

public class HelloCalaisModel {
    public static void main(String[] args) {
        // read RDF file from disk
        String path2file = "<path-to-TRIT-rdf-output>";
        CalaisModel calaisModel = CalaisModelCreator.readFile(path2file,
            Engine.Format.RDF);

        // get all types found in RDF
        Set<String> allTypes = calaisModel.getAllTypes();
        for (String type : allTypes) {
            System.out.println(type);
        }
    }
}
```

The following is an example response:

```
http://s.opencalais.com/1/type/em/e/Company
http://s.opencalais.com/1/type/em/r/Quotation
http://s.opencalais.com/1/type/cat/DocCat
http://s.opencalais.com/1/type/tag/SocialTag
http://s.opencalais.com/1/type/em/e/Technology
http://s.opencalais.com/1/type/tag/Confidence
http://s.opencalais.com/1/type/er/Company
...
```

The response is a list of the metadata types contained in the RDF output.

The RDF file can include any of a number of metadata types, all of which are described in the API User Guide.

Note that the above example includes two important metadata types: `em/e/Company`, and `er/Company`.

All Entity Markup tags begin with the prefix: `http://s.opencalais.com/1/type/em/e/`

For example, a phrase that TRIT identifies as a company is extracted as an `em/e/Company` tag, which is an entity markup tag of the type `Company`: `http://s.opencalais.com/1/type/em/e/Company`

TRIT attempts to map extracted entities to the matching entity and unique ID in the relevant Thomson Reuters dataset. If the mapping is successful, a Disambiguation (Resolution) tag is added to the RDF output file.

All disambiguation tags begin with the prefix: `http://s.opencalais.com/1/type/er/`

For example, if an extracted company is mapped to a company in the Thomson Reuters data, a `Company` disambiguation tag is output: `http://s.opencalais.com/1/type/er/Company`

3.2 Example 2: Retrieving all the Objects of a Specified Type

You can use the CalaisModel method “getCalaisObjectByType” to retrieve from the RDF output all the objects of a specified type and then use the resulting object list to extract and analyze the object data.

The following code snippet retrieves all of the company extractions (all of the em/e/Company tags) and their confidence scores. (The Confidence score indicates the likelihood that the extracted company is indeed a company.)

```
List<CalaisObject> extractedCompanies =
    calaisModel.getCalaisObjectByType("http://s.opencalais.com/1/type/em/e/Company");
for (CalaisObject extractedCompany : extractedCompanies) {
    Map<String, List<String>> literals = extractedCompany.getLiterals();
    List<String> companyName = literals.get("http://s.opencalais.com/1/pred/name");
    List<String> confidenceLevel = literals.get("http://s.opencalais.com/1/pred/
confidencelevel");
    String line2print = String.format("Company [%s] is extracted with [%s] confidence",
        companyName.get(0), confidenceLevel.get(0));
    System.out.println(line2print);
}
```

The response is a list of extracted companies and confidence scores:

```
Company [C. J. Heo] is extracted with [0.765] confidence
Company [Huawei Technologies Co Ltd] is extracted with [0.993] confidence
Company [Qualcomm] is extracted with [0.904] confidence
Company [Nokia Corp] is extracted with [0.957] confidence
Company [Ericsson] is extracted with [0.893] confidence
Company [Intel Corp.] is extracted with [0.999] confidence
Company [Reuters] is extracted with [0.789] confidence
Company [Verizon Communications Inc] is extracted with [0.999] confidence
Company [Samsung Electronics] is extracted with [0.974] confidence
```

3.3 Example 3: Filtering the Object List and Retrieving Specified Related Data

Now let's pull it all together with a more complex example.

The following example code retrieves from the RDF output, specified details about companies that were extracted with confidence s greater than 0.8, and disambiguated with scores greater than 0.85.

```
public static void main(String[] args) {
    final CalaisModel calaisModel = CalaisModelCreator.readFile("path\\OutputRDFFile.xml", Format.RDF);
    List<CalaisObject> extractedCompanies =
    calaisModel.getCalaisObjectByType("http://s.opencalais.com/1/type/em/e/Company");
    for (CalaisObject extractedCompany : extractedCompanies) {
        Map<String, List<String>> literals = extractedCompany.getLiterals();
        List<String> confidencelevel = literals.get("http://s.opencalais.com/1/pred/confidencelevel");
        //get only companies with confidencelevel greater then 0.8
        if(Double.parseDouble(confidencelevel.get(0)) > 0.8){
            Map<String, List<CalaisObject>> backReferencesSubjectField =
            extractedCompany.getBackReferencesByFieldName("http://s.opencalais.com/1/pred/subject");
            //get the list of resolved (disambiguated) companies
            List<CalaisObject> resolvedCompanies =
            backReferencesSubjectField.get("http://s.opencalais.com/1/type/er/Company");
            for (CalaisObject resolvedCompany : resolvedCompanies) {
                Map<String, List<String>> resolvedCompanyLiterals = resolvedCompany.getLiterals();
                List<String> score =
                resolvedCompanyLiterals.get("http://s.opencalais.com/1/pred/score");
                if(Double.parseDouble(score.get(0)) > 0.85){
                    List<String> companyName = literals.get("http://s.opencalais.com/1/pred/name");
                    List<String> permId =
                    resolvedCompanyLiterals.get("http://s.opencalais.com/1/pred/permid");
                    List<String> ticker =
                    resolvedCompanyLiterals.get("http://s.opencalais.com/1/pred/ticker");
                    List<String> commonname =
                    resolvedCompanyLiterals.get("http://s.opencalais.com/1/pred/commonname");
                    List<String> primaryric =
                    resolvedCompanyLiterals.get("http://s.opencalais.com/1/pred/primaryric");
                    //get the InstanceInfo details
                    List<CalaisObject> backReferencesInstanceInfo =
                    backReferencesSubjectField.get("http://s.opencalais.com/1/type/sys/InstanceInfo");
                    for (CalaisObject instanceInfo : backReferencesInstanceInfo) {
                        Map<String, List<String>> instanceInfoLiterals = instanceInfo.getLiterals();
                        List<String> exact =
                        instanceInfoLiterals.get("http://s.opencalais.com/1/pred/exact");
                        List<String> offset =
                        instanceInfoLiterals.get("http://s.opencalais.com/1/pred/offset");
                        System.out.println(String.format("found the company:%s, at the
                        offset:%s.\ncompany details:" +
                        "\ncompany name:%s,\npermId:%s,\nticker:%s,\ncommon
                        name:%s,\nprimaryRic:%s",exact, offset, companyName, permId, ticker, commonname,primaryric));
                    }
                }
            }
        }
    }
}
```

© 2017 Thomson Reuters. All rights reserved. Republication or redistribution of Thomson Reuters content, including by framing or similar means, is prohibited without the prior written consent of Thomson Reuters. 'Thomson Reuters' and the Thomson Reuters logo are registered trademarks and trademarks of Thomson Reuters and its affiliated companies.

Date of Issue: 06 June 2017



THOMSON REUTERS