National Cancer Institute U.S. National Institutes of Health | www.cancer.gov

caBIG® Knowledge Center
A part of the Enterprise Support Network

Log in create account account help

Home Knowledge Centers Discussion Forums Bugs/Feature Requests Development Code Repository

page discussion view source history

LexEVS 5.x Migration EVS API to LexEVS API

Vocabulary administration > LexEVS 5.x Migration Guide > LexEVS 5.x Migration LexEVS model > LexEVS 5.x Migration LexEVS database > LexEVS 5.x Migration EVS API to LexEVS API

Contents [hide]

vocabkc contents
- Main Page
- What's New
- Forums
- Bugzilla
- Code Repository
- Feedback
- Contact Us

tools
- LexBIG/LexEVS
- LexWiki
- NCI Protégé
- Related Tools and Models

projects
- LexAjax
- LexGrid
- Cancer Data Standards Repository (caDSR)
- Common Terminology Criteria for Adverse Events (CTCAE)
- Open Health Natural Language Processing (OHNLP) Consortium
- Ontology Development and Information Extraction (ODIE)

semantic infrastructure
- SI Main Page
- Initiatives
- Requirements

other resources
- Library of Documents
- Documentation and Training for Tools
- Index of Terminologies
- Standards and Standards Influencing Organizations
- Outreach

external links
- VCDE Workspace
- caBIG® Community Website
- caBIG® Support Service Providers

help
- Editing Wiki Pages

- Editing Forum Posts
- Contact Us

search

Search

toolbox

- What links here
- Related changes
- Upload file
- Special pages
- Printable version
- Permanent link
- Print as PDF

Introduction

This document is a section of the Migration Guide.

Purpose of this document

The transition from EVS API to LexEVS API will require the use of different methods to accomplish the same function as was previously provided in EVS API. This document will identify each deprecated class and provide the alternative LexEVS API.

Classes no longer available

As a result of the deprecation of the EVS API, the following classes are no longer available:

- EVSQuery and EVSQueryImpl
- EVSQueryDAOImpl - Not a public API.
- EVSWSDAOImpl - Not a public API.
- EVSWSQuery - Not a public API.
- DLBAdapter - Not a public API.
- DLBWrapper (DLBWrapper is extended by DLBAdapter and it is not been used anywhere else) - Not a public API.
- EVSApplicationService and EVSApplicationServiceImpl

EVSApplicationService and LexEVS Counterparts

EVSApplicationService

evsSearch and search

These methods are replaced either by the LexEVS API or the LexEVS caCORE SDK Data Service 'search' method. This uses the standard caCORE SDK API. More information about the caCORE SDK can be obtained on the caCORE SDK site.

EVSQuery Methods and LexEVS Counterparts

EVSQuery

getTree

All 'getTree' functionality is replaced by the LexEVS `CodedNodeGraph` API. For example:

```
 public void getTree(String vocabularyName, String rootCode, boolean
direction, boolean isaFlag, int attributes, int levels, Vector roles);
```

is replaced by:

```
 CodedNodeGraph cng = lexevsService.getNodeGraph(String codingScheme,
CodingSchemeVersionOrTag, versionOrTag, String relationContainerName);
 ResolvedConceptReference[] rcr = cng.resolveAsList(ConceptReference
graphFocus, boolean resolveForward,
                                                    boolean
resolveBackward, int resolveCodedEntryDepth, int
resolveAssociationDepth,
                                                    LocalNameList
propertyNames, PropertyType[] propertyTypes, SortOptionList
sortOptions,
                                                    int
maxToReturn).getResolvedConceptReference();
```

Also, various methods in `LexBIGServiceConvenienceMethods` can be used to show hierarchies. See the following methods:

- getHierarchyRoots
- getHierarchyRootSet
- getHierarchyLevelNext
- getHierarchyLevelPrev
- getHierarchyPathToRoot

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.BuildTreeForCode
- org.LexGrid.LexBIG.example.ListHierarchy
- org.LexGrid.LexBIG.example.ListHierarchyByCode
- org.LexGrid.LexBIG.example.ListHierarchyPathToRoot

searchDescLogicConcepts

LexEVS provides many ways to restrict the result of a query. The method 'searchDescLogicConcepts' searches for matches based on a text String. To conduct similar queries using LexEVS, use the `CodedNodeSet` API.

Obtain a CodedNodeSet from LexEVS:

```
 CodedNodeSet nodes = lexevsService.getNodeSet(String codingScheme,
CodingSchemeVersionOrTag versionOrTag, LocalNameList entityTypes);
```

Once established, the CodedNodeSet can be further restricted using the various 'restrict' methods in the `CodedNodeSet` API.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.SoundsLike
- org.LexGrid.LexBIG.example.FindCodesForDescription

getConceptWithPropertyMatching

See example above. For Property-specific matching, see the following method in the `CodedNodeSet` API

- restrictToProperties

This will ensure that each of the results will have at least one Propety that matches the supplied criteria.

isSubConcept

In LexEVS, use the `CodedNodeGraph` API to find the immediate relations of a Concept. For instance:

```
 CodedNodeGraph cng = lexevsService.getNodeGraph(String codingScheme,
CodingSchemeVersionOrTag versionOrTag, String relationContainerName);
 ResolvedConceptReference[] rcr = cng.resolveAsList(ConceptReference
graphFocus, boolean resolveForward, boolean resolveBackward,
                                                   int
resolveCodedEntryDepth, int resolveAssociationDepth, LocalNameList
propertyNames,
                                                   PropertyType[]
propertyTypes, SortOptionList sortOptions, int
maxToReturn).getResolvedConceptReference();
```

Set the `ConceptReference graphFocus` to the desired code, this will focus the Graph. The check the relationships.

Alternatively, use the CodedNodeGraph API method 'areCodesRelated'

```
 Boolean areCodesRelated(NameAndValue association, ConceptReference
sourceCode, ConceptReference targetCode, boolean directOnly)
```

isRetired

Use the LexBIGServiceConvenienceMethods API method 'isCodeRetired' method.

getDescendants

Use the LexEVS HistoryService API

Obtain a the HistoryService API as follows:

```
 HistoryService historySvc = lexevsService.getHistoryService
(StringcodingSchemeName);
```

To get the decendents of a given code, use the 'getDecendants' method:

```
 NCIChangeEventList changeEventList = historySvc.getDescendants
(ConceptReference conceptReference);
```

getPropertyValues

Through the ***Entity*** class, all Properties (Presentations, Definitions, etc) are available.

- Entity
- To find the 'Presentations' of a given Entity:

```
   Entity entity = ....;
   Presentation[] presentations = entity.getPresentation();
```

- To find the 'Definitions' of a given Entity:

```
   Entity entity = ....;
   Definition[] definitions = entity.getDefinition();
```

- To find the 'Comments' of a given Entity:

```
   Entity entity = ....;
   Comment[] comments= entity.getComment();
```

- To find the non-classified Properties of a given Entity:

```
   Entity entity = ....;
   Property[] properties = entity.getProperties();
```

getAncestors

Use the LexEVS HistoryService API

Obtain a the HistoryService API as follows:

```
 HistoryService historySvc = lexevsService.getHistoryService
(StringcodingSchemeName);
```

To get the ancestors of a given code, use the 'getDecendants' method:

```
 NCIChangeEventList changeEventList = historySvc.getAncestors
(ConceptReference conceptReference);
```

getSubConcepts

Use the CodedNodeGraph API to find the immediate relations of a Concept. For instance:

```
 CodedNodeGraph cng = lexevsService.getNodeGraph(String codingScheme,
CodingSchemeVersionOrTag versionOrTag, String relationContainerName);
 ResolvedConceptReference[] rcr = cng.resolveAsList(ConceptReference
graphFocus, boolean resolveForward, boolean resolveBackward,
                                                   int
resolveCodedEntryDepth, int resolveAssociationDepth, LocalNameList
propertyNames,
                                                   PropertyType[]
propertyTypes, SortOptionList sortOptions, int
maxToReturn).getResolvedConceptReference();
```

Focus the 'graphFocus' on the desired Concept to see relationships from a given Concept.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getSuperConcepts

See above 'getSubConcepts' -- setting your resolve direction (boolean resolveForward, boolean resolveBackward) will determine if sub-concepts or super-concepts are resolved.

getPropertiesByConceptCode

To find the Properties of a given code, for example, code 'C1234' in the 'NCI Thesaurus' ontology - first obtain a 'CodedNodeSet' from the 'NCI Thesaurus' ontology:

```
 ResolvedConceptReferenceList cns = lbSvc.getCodingSchemeConcepts("NCI
Thesaurus", null);
```

Next, restrict that to the desired Code ('C1234' in this example):

```
 //First create a ConceptReferenceList to describe the Concept to search for.
 //In this example we use the helper class 'ConvenienceMethods'.
 ConceptReferenceList crefs =
ConvenienceMethods.createConceptReferenceList(new String[]
{ "C1234"}, "NCI Thesaurus");

 //Next, restrice the CodedNodeSet.
 cns.restrictToCodes(crefs);
```

Lastly, resolve the match.

```
 ResolvedConceptReferenceList matches = cns.resolveToList(null, null,
null, 1);
```

To view the Properties, see 'getPropertyValues' above.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getVocabularyNames

Use the 'LexBIGService' API method 'getSupportedCodingSchemes' - and extract the Names (local name, registered name, etc...) as needed.

getAllVocabularies

Use the 'LexBIGService' API method 'getSupportedCodingSchemes'.

getVocabularyByName

Use 'LexBIGService' API method 'resolveCodingScheme'.

getVocabularyVersion

Use 'LexBIGService' API method 'resolveCodingScheme' and extract the 'representsVersion' attribute from the Resulting CodingScheme.

getConceptEditAction

Use the LexEVS HistoryService API

Obtain a the HistoryService API as follows:

```
 HistoryService historySvc = lexevsService.getHistoryService
(StringcodingSchemeName);
```

To get the 'EditActionList' of a given code, use the 'getEditActionList' method:

```
 NCIChangeEventList changeEventList = historySvc.getEditActionList(ConceptReference conceptReference,
CodingSchemeVersion codingSchemeVersion);
```

getRootConcepts

Use 'LexBIGServiceConvenienceMethods' API method 'getHierarchyRoots' or 'getHierarchyRootSet'

searchSourceByCode

Use 'CodedNodeSet' API - adding a 'restrictToCodes' restriction.

To find the a given code, for example, code 'C1234' in the 'NCI Thesaurus' ontology - first obtain a 'CodedNodeSet' from the 'NCI Thesaurus' ontology:

```
 ResolvedConceptReferenceList cns = lbSvc.getCodingSchemeConcepts("NCI
Thesaurus", null);
```

Next, restrict that to the desired Code ('C1234' in this example):

```
 //First create a ConceptReferenceList to describe the Concept to
search for.
 //In this example we use the helper class 'ConvenienceMethods'.
 ConceptReferenceList crefs = ConvenienceMethods.createConceptReferenceList
(new String[]
{ "C1234"}, "NCI Thesaurus");

 //Next, restrice the CodedNodeSet.
 cns.restrictToCodes(crefs);
```

Lastly, resolve the match.

```
 ResolvedConceptReferenceList matches = cns.resolveToList(null, null,
null, 1);
```

searchSourceByAtomCode

In the NCI MetaThesaurus, fidning the 'source' of an 'Atom' is equivalent to finding the 'source' of a given Property of an Entity. Each CUI (which is equivalent to an Entity in LexEVS) may contain several Presentation Properties (Atoms or AUI's of that CUI). Each of these Presentation Properties is Qualified by a 'source-code' Qualifier, which reflects the code of this Atom in its original source, and a 'source' qualifier, which states the source itself that this Atom came from.

getMetaConceptNameByCode

To find the Properties of a given code, for example, code 'C1234567' in the 'NCI MetaThesaurus' ontology - first obtain a 'CodedNodeSet' from the 'NCI MetaThesaurus' ontology:

```
 ResolvedConceptReferenceList cns = lbSvc.getCodingSchemeConcepts("NCI
MetaThesaurus", null);
```

Next, restrict that to the desired Code ('C1234567' in this example):

```
 //First create a ConceptReferenceList to describe the Concept to
search for.
 //In this example we use the helper class 'ConvenienceMethods'.
 ConceptReferenceList crefs =
ConvenienceMethods.createConceptReferenceList(new String[]
{ "C1234567"}, "NCI MetaThesaurus");

 //Next, restrice the CodedNodeSet.
 cns.restrictToCodes(crefs);
```

Lastly, resolve the match.

```
 ResolvedConceptReferenceList matches = cns.resolveToList(null, null,
null, 1);
```

To see the name of the code, use 'getEntityDescription' on the resulting ResolvedConceptReference. The 'EntityDescription' will always be equal to the Preferred Presentation in the Default Language.

getMetaSources

Use the 'LexBIGService' API method 'getMappings'. Extract for this the Supported Association Qualifiers using the 'getSupportedSource' method. Note: This can be applied to any Coding Scheme, not just the NCI MetaThesaurus.

getChildren

Use the 'CodedNodeGraph' API.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getParent

Use the 'CodedNodeGraph' API.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getBroaderConcepts

Use the CodedNodeGraph API to find the immediate relations of a Concept. Resolve forward or backwards based on the hierarchy structure of the ontology.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getNarrowerConcepts

Use the CodedNodeGraph API to find the immediate relations of a Concept. Resolve forward or backwards based on the hierarchy structure of the ontology.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getRelatedConcepts

Use the CodedNodeGraph API to find the immediate relations of a Concept.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

containsInverseRole

Use the LexBIGServiceConvenienceMethods API method 'isReverseName' method.

containsRole

Use the LexBIGServiceConvenienceMethods API method 'isForwardName' method.

getAllAssociationTypes

Use the 'LexBIGService' API method 'getMappings'. Extract for this the Supported Associations using the 'getSupportedAssociation' method.

getAllConceptAssociationQualifierTypes

Use the 'LexBIGService' API method 'getMappings'. Extract for this the Supported Association Qualifiers using the 'getSupportedAssociationQualifier' method.

getAllConceptAssociationTypes

Use the 'LexBIGService' API method 'getMappings'. Extract for this the Supported Associations using the 'getSupportedAssociation' method.

getAllConceptPropertyQualifierTypes

Use the 'LexBIGService' API method 'getMappings'. Extract for this the Supported Property Qualifiers using the 'getSupportedPropertyQualifier' method.

getAllConceptPropertyTypes

Use the 'LexBIGService' API method 'getMappings'. Extract for this the Supported Properties using the 'getSupportedProperty' method.

getAllLicenses

Use the 'LexBIGService' API method 'resolveCodingSchemeCopyright'. To get the Copyright for every loaded ontology, do this for each one.

getAllPropertyTypes

Use the 'LexBIGService' API method 'getMappings'. Extract for this the Supported Properties using the 'getSupportedProperty' method.

getAllQualifierTypes

Use the 'LexBIGService' API method 'getMappings'. Extract for this the Supported Property Qualifiers using the 'getSupportedPropertyQualifier' method.

getAllRoleNames

Use the 'LexBIGService' API method 'resolveCodingScheme'. Once the CodingScheme Object is obtained, use the method 'getRelations'.

getAllSubConceptCodes

Use the CodedNodeGraph API to find the immediate relations of a Concept.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getAllSynonymTypes

Use the 'LexBIGService' API method 'getMappings'. Extract for this the Supported Associations using the 'getSupportedAssociation' method. NOTE: Different ontologies may describe their 'Synonym' relations differently.

getAllTermAssociationQualifierTypes

Use the 'LexBIGService' API method 'getMappings'. Extract for this the Supported Association Qualifiers using the 'getSupportedAssociationQualifier' method.

getAllTermPropertyQualifierTypes

Use the 'LexBIGService' API method 'getMappings'. Extract for this the Supported Property Qualifiers using the 'getSupportedPropertyQualifier' method.

getAllTermPropertyTypes

Use the 'LexBIGService' API method 'getMappings'. Extract for this the Supported Property using the 'getSupportedProperty' method.

getParentConcepts

Use the CodedNodeGraph API to find the immediate relations of a Concept.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getChildConcepts

Use the CodedNodeGraph API to find the immediate relations of a Concept.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

hasParents

Use the CodedNodeGraph API to find the immediate relations of a Concept.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

hasChildren

Use the CodedNodeGraph API to find the immediate relations of a Concept.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getDescLogicConcept

A 'DescLogicConcept' can be thought of as an 'Entity' in LexEVS. To obtain an Entity, use the CodedNodeSet API, restricting the query as necessary.

For instance, a 'DescLogicConcept' with a code of 'C1234' can be queried for by the example below. The example will return a ResolvedConceptReference and ultimately an Entity, but is functionally the same as searching for a DescLogicConcept.

To find the a given code, for example, code 'C1234' in the 'NCI Thesaurus' ontology - first obtain a 'CodedNodeSet' from the 'NCI Thesaurus' ontology:

```
 ResolvedConceptReferenceList cns = lbSvc.getCodingSchemeConcepts("NCI
Thesaurus", null);
```

Next, restrict that to the desired Code ('C1234' in this example):

```
 //First create a ConceptReferenceList to describe the Concept to
search for.
 //In this example we use the helper class 'ConvenienceMethods'.
 ConceptReferenceList crefs =
ConvenienceMethods.createConceptReferenceList(new String[]
{ "C1234"}, "NCI Thesaurus");

 //Next, restrice the CodedNodeSet.
 cns.restrictToCodes(crefs);
```

Lastly, resolve the match.

```
 ResolvedConceptReferenceList matches = cns.resolveToList(null, null,
null, 1);
```

getHistoryRecords

Use the HistoryService API 'getBaselines' and specify the required Data range.

Obtain a the HistoryService API as follows:

```
 HistoryService historySvc = lexevsService.getHistoryService
(StringcodingSchemeName);
```

Use the 'getBaselines' method:

```
 SystemReleaseList systemReleaseList = historySvc.getBaselines(Date
releasedAfter, Date releasedBefore);
```

getHistoryStartDate

Use the HistoryService API method 'getEarliestBaseline';

Obtain a the HistoryService API as follows:

```
 HistoryService historySvc = lexevsService.getHistoryService
(StringcodingSchemeName);
```

Use the 'getEarliestBaseline' method:

```
 SystemRelease systemRelease = historySvc.getEarliestBaseline();
```

getHistoryEndDate

Use the HistoryService API method 'getLatestBaseline';

Obtain a the HistoryService API as follows:

```
 HistoryService historySvc = lexevsService.getHistoryService
(StringcodingSchemeName);
```

Use the 'getLatestBaseline' method:

```
 SystemRelease systemRelease = historySvc.getLatestBaseline();
```

getCodeActionChildren

Use the HistoryService API method 'getDescendants'; Obtain a the HistoryService API as follows:

```
 HistoryService historySvc = lexevsService.getHistoryService
(StringcodingSchemeName);
```

Use the 'getDescendants' method:

```
 NCIChangeEventList changeEventList = historySvc.getDescendants
(ConceptReference conceptReference);
```

getCodeActionParents

Use the HistoryService API method 'getAncestors';

Obtain a the HistoryService API as follows:

```
 HistoryService historySvc = lexevsService.getHistoryService
(StringcodingSchemeName);
```

Use the 'getDescendants' method:

```
 NCIChangeEventList changeEventList = historySvc.getAncestors
(ConceptReference conceptReference);
```

getAssociationCollectionbyCode

Use the CodedNodeGraph API to find the immediate relations of a Concept. For instance:

```
 CodedNodeGraph cng = lexevsService.getNodeGraph(String codingScheme,
CodingSchemeVersionOrTag versionOrTag, String relationContainerName);
 ResolvedConceptReference[] rcr = cng.resolveAsList(ConceptReference
graphFocus, boolean resolveForward, boolean resolveBackward,
                                                    int
resolveCodedEntryDepth, int resolveAssociationDepth, LocalNameList
propertyNames,
                                                    PropertyType[]
propertyTypes, SortOptionList sortOptions, int
maxToReturn).getResolvedConceptReference();
```

Focus the 'graphFocus' on the desired Concept to see relationships from a given Concept.

Once focused and resolved, use the 'getSourceOf' or 'getTargetOf' methods on the ResolvedConceptReference to find the Associations of a given Code.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getSemanticTypeCollectionbyCui

Use 'CodedNodeSet' API - adding a 'restrictToCodes' restriction.

Note that a CUI is simply a reference to a Code in the NCI MetaThesaurus ontology.

To find the a given code, for example, code 'C1234' in the 'NCI Thesaurus' ontology - first obtain a 'CodedNodeSet' from the 'NCI Thesaurus' ontology:

```
 ResolvedConceptReferenceList cns = lbSvc.getCodingSchemeConcepts("NCI
Thesaurus", null);
```

Next, restrict that to the desired Code ('C1234' in this example):

```
 //First create a ConceptReferenceList to describe the Concept to
search for.
 //In this example we use the helper class 'ConvenienceMethods'.
 ConceptReferenceList crefs =
ConvenienceMethods.createConceptReferenceList(new String[]
{ "C1234"}, "NCI Thesaurus");

 //Next, restrice the CodedNodeSet.
 cns.restrictToCodes(crefs);
```

Lastly, resolve the match.

```
 ResolvedConceptReferenceList matches = cns.resolveToList(null, null,
null, 1);
```

Once the ResolvedConceptReference has been obtained, extract the desired Properties and inspect the Qualifiers for the Semantic Type.

getQualifierCollectionbyName

Use 'CodedNodeSet' API - adding a 'restrictToCodes' restriction.

Note that a CUI is simply a reference to a Code in the NCI MetaThesaurus ontology.

To find the a given code, for example, code 'C1234' in the 'NCI Thesaurus' ontology - first obtain a 'CodedNodeSet' from the 'NCI Thesaurus' ontology:

```
 ResolvedConceptReferenceList cns = lbSvc.getCodingSchemeConcepts("NCI
Thesaurus", null);
```

Next, restrict that to the desired Code ('C1234' in this example):

```
 //First create a ConceptReferenceList to describe the Concept to
search for.
 //In this example we use the helper class 'ConvenienceMethods'.
 ConceptReferenceList crefs =
ConvenienceMethods.createConceptReferenceList(new String[]
{ "C1234"}, "NCI Thesaurus");

 //Next, restrice the CodedNodeSet.
 cns.restrictToCodes(crefs);
```

Lastly, resolve the match.

```
 ResolvedConceptReferenceList matches = cns.resolveToList(null, null,
null, 1);
```

Once the ResolvedConceptReference has been obtained, extract the desired Properties and inspect the Qualifiers.

NOTE: Associations between codes may also have Qualifiers.

getAtomCollectionbyCui

In LexEVS, a NCI MetaThesaurus CUI is represented by an Entity (with the CUI being the code for that Entity). Atoms of that CUI are represented by 'Presentation'(s) of the Entity.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getSynonymCollectionbyCui

Use 'CodedNodeGraph' API - restricting to Synonym Associations. Note: Each ontology may describe their Synonym Associations differently.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getSourceCollectionbyCui

Use 'CodedNodeSet' API - adding a 'restrictToCodes' restriction.

Note that a CUI is simply a reference to a Code in the NCI MetaThesaurus ontology.

To find the a given code, for example, code 'C1234' in the 'NCI Thesaurus' ontology - first obtain a 'CodedNodeSet' from the 'NCI Thesaurus' ontology:

```
 ResolvedConceptReferenceList cns = lbSvc.getCodingSchemeConcepts("NCI
Thesaurus", null);
```

Next, restrict that to the desired Code ('C1234' in this example):

```
 //First create a ConceptReferenceList to describe the Concept to
search for.
 //In this example we use the helper class 'ConvenienceMethods'.
 ConceptReferenceList crefs =
ConvenienceMethods.createConceptReferenceList(new String[]
{ "C1234"}, "NCI Thesaurus");

 //Next, restrice the CodedNodeSet.
 cns.restrictToCodes(crefs);
```

Lastly, resolve the match.

```
 ResolvedConceptReferenceList matches = cns.resolveToList(null, null,
null, 1);
```

Once the ResolvedConceptReference has been obtained, extract the desired Properties and inspect Source using 'getSource'.

getSemanticTypeCollectionbyCui

Use 'CodedNodeSet' API - adding a 'restrictToCodes' restriction.

Note that a CUI is simply a reference to a Code in the NCI MetaThesaurus ontology.

To find the a given code, for example, code 'C1234' in the 'NCI Thesaurus' ontology - first obtain a 'CodedNodeSet' from the 'NCI Thesaurus' ontology:

```
 ResolvedConceptReferenceList cns = lbSvc.getCodingSchemeConcepts("NCI
Thesaurus", null);
```

Next, restrict that to the desired Code ('C1234' in this example):

```
 //First create a ConceptReferenceList to describe the Concept to
search for.
 //In this example we use the helper class 'ConvenienceMethods'.
 ConceptReferenceList crefs =
ConvenienceMethods.createConceptReferenceList(new String[]
{ "C1234"}, "NCI Thesaurus");

 //Next, restrice the CodedNodeSet.
 cns.restrictToCodes(crefs);
```

Lastly, resolve the match.

```
 ResolvedConceptReferenceList matches = cns.resolveToList(null, null,
null, 1);
```

Once the ResolvedConceptReference has been obtained, extract the desired Properties and inspect the Semantic Types. Semantic Types are held as Qualifiers to the Properties of an Entity.

getSourcebyDefinition

Use 'CodedNodeSet' API - adding a 'restrictToCodes' restriction.

To find a given code, for example, code 'C1234' in the 'NCI Thesaurus' ontology - first obtain a 'CodedNodeSet' from the 'NCI Thesaurus' ontology:

```
 ResolvedConceptReferenceList cns = lbSvc.getCodingSchemeConcepts("NCI
Thesaurus", null);
```

Next, restrict that to the desired Code ('C1234' in this example):

```
 //First create a ConceptReferenceList to describe the Concept to
search for.
 //In this example we use the helper class 'ConvenienceMethods'.
 ConceptReferenceList crefs =
ConvenienceMethods.createConceptReferenceList(new String[]
{ "C1234"}, "NCI Thesaurus");

 //Next, restrice the CodedNodeSet.
 cns.restrictToCodes(crefs);
```

Lastly, resolve the match.

```
 ResolvedConceptReferenceList matches = cns.resolveToList(null, null,
null, 1);
```

Once the ResolvedConceptReference has been obtained, using 'getDefinition', extract the Definition Collection from the Entity. Then extract the source using 'getSource'

getDefinitionCollectionbyCui

Use 'CodedNodeSet' API - adding a 'restrictToCodes' restriction.

To find the a given code, for example, code 'C1234' in the 'NCI Thesaurus' ontology - first obtain a 'CodedNodeSet' from the 'NCI Thesaurus' ontology:

```
 ResolvedConceptReferenceList cns = lbSvc.getCodingSchemeConcepts
("NCI Thesaurus", null);
```

Next, restrict that to the desired Code ('C1234' in this example):

```
 //First create a ConceptReferenceList to describe the Concept to search for.
 //In this example we use the helper class 'ConvenienceMethods'.
 ConceptReferenceList crefs = ConvenienceMethods.createConceptReferenceList
(new String[]
{ "C1234"}, "NCI Thesaurus");

 //Next, restrice the CodedNodeSet.
 cns.restrictToCodes(crefs);
```

Lastly, resolve the match.

```
 ResolvedConceptReferenceList matches = cns.resolveToList(null, null,
null, 1);
```

Once the ResolvedConceptReference has been obtained, use the 'getReferencedEntry' method to obtain the actual Entity. Using the 'getDefinition', extract the Definition Collection from the Entity.

getPropertyCollectionbyName

To find the Properties of a given code, for example, a code with a name of 'Heart' in the 'NCI Thesaurus' ontology - first obtain a 'CodedNodeSet' from the 'NCI Thesaurus' ontology:

```
 ResolvedConceptReferenceList cns = lbSvc.getCodingSchemeConcepts
("NCI Thesaurus", null);
```

Next, restrict that to the desired Code Name ('Heart' in this example): Note: To match the String 'Heart' exactly, use the search algorithm 'exactMatch'.

```
//Next, restrice the CodedNodeSet.
  cns = cns.restrictToMatchingDesignations("Heart", null, "exactMatch", null);
```

Lastly, resolve the match.

```
ResolvedConceptReferenceList matches = cns.resolveToList(null, null, null, 1);
```

To view the Properties, see 'getPropertyValues' above.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getPropertyCollectionbyCode

To find the Properties of a given code, for example, code 'C1234' in the 'NCI Thesaurus' ontology - first obtain a 'CodedNodeSet' from the 'NCI Thesaurus' ontology:

```
 ResolvedConceptReferenceList cns = lbSvc.getCodingSchemeConcepts
("NCI Thesaurus", null);
```

Next, restrict that to the desired Code ('C1234' in this example):

```
 //First create a ConceptReferenceList to describe the Concept to search for.
 //In this example we use the helper class 'ConvenienceMethods'.
 ConceptReferenceList crefs = ConvenienceMethods.createConceptReferenceList
(new String[]
{ "C1234"}, "NCI Thesaurus");

 //Next, restrice the CodedNodeSet.
 cns.restrictToCodes(crefs);
```

Lastly, resolve the match.

```
 ResolvedConceptReferenceList matches = cns.resolveToList(null, null,
null, 1);
```

To view the Properties, see 'getPropertyValues' above.

- For examples, see the LexEVS Example classes
  - org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

fetchPropertyCollectionByCodes

See 'getPropertyCollectionbyCode;

getRoleCollectionbyCode

Use the CodedNodeGraph API to find the immediate relations of a Concept.

If a desired Relations Collection is requested (for example, 'roles'), the 'relationContainerName' may be restricted. For instance:

```
 //Restrict the 'relationContainerName' to the desired container.
NOTE: These containers are
 //ontology specific--each ontology defines its own relation container names.
 CodedNodeGraph cng = lexevsService.getNodeGraph(String codingScheme,
CodingSchemeVersionOrTag versionOrTag, String relationContainerName);
 ResolvedConceptReference[] rcr = cng.resolveAsList(ConceptReference
graphFocus, boolean resolveForward, boolean resolveBackward,
                                           int resolveCodedEntryDepth, int
resolveAssociationDepth, LocalNameList propertyNames,
                                          PropertyType[] propertyTypes,
SortOptionList sortOptions, int maxToReturn).getResolvedConceptReference();
```

Focus the 'graphFocus' on the desired Concept to see relationships from a given Concept.

Once focused and resolved, use the 'getSourceOf' or 'getTargetOf' methods on the ResolvedConceptReference to find the Associations of a given Code.

- For examples, see the LexEVS Example classes
  - org.LexGrid.LexBIG.example.FindRelatedCodes
  - org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getInverseRoleCollectionbyCode

Use the CodedNodeGraph API to find the immediate relations of a Concept. To find the Inverse Roles, restricting the Relations Container may be necessary (see getRoleCollectionbyCode above). Depending on how the onotology defines an 'inverse' role or association, these can be restricted as well.

For instance:

```
 CodedNodeGraph cng = lexevsService.getNodeGraph(String codingScheme,
CodingSchemeVersionOrTag versionOrTag, String relationContainerName);
 ResolvedConceptReference[] rcr = cng.resolveAsList(ConceptReference
graphFocus, boolean resolveForward, boolean resolveBackward,
                                         int resolveCodedEntryDepth, int
resolveAssociationDepth, LocalNameList propertyNames,
                                         PropertyType[] propertyTypes,
SortOptionList sortOptions, int maxToReturn).getResolvedConceptReference();
```

Focus the 'graphFocus' on the desired Concept to see relationships from a given Concept.

Once focused and resolved, use the 'getSourceOf' or 'getTargetOf' methods on the ResolvedConceptReference to find the Associations of a given Code.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getInverseAssociationCollectionbyCode

Use the CodedNodeGraph API to find the immediate relations of a Concept.

To find the Inverse Collections, restricting the Relations Container may be necessary (see getInverseRoleCollectionbyCode above). Depending on how the onotology defines an 'inverse' role or association, these can be restricted as well.

For instance:

```
 CodedNodeGraph cng = lexevsService.getNodeGraph(String codingScheme,
CodingSchemeVersionOrTag versionOrTag, String relationContainerName);
 ResolvedConceptReference[] rcr = cng.resolveAsList(ConceptReference
graphFocus, boolean resolveForward, boolean resolveBackward,
                                       int resolveCodedEntryDepth, int
resolveAssociationDepth, LocalNameList propertyNames,
                                       PropertyType[] propertyTypes,
SortOptionList sortOptions, int maxToReturn).getResolvedConceptReference();
```

Focus the 'graphFocus' on the desired Concept to see relationships from a given Concept.

Once focused and resolved, use the 'getSourceOf' or 'getTargetOf' methods on the ResolvedConceptReference to find the Associations of a given Code.

For examples, see the following LexEVS Example classes:

- org.LexGrid.LexBIG.example.FindRelatedCodes
- org.LexGrid.LexBIG.example.FindPropsAndAssocForCode

getHasParentsbyCode

Either:

- Use the 'LexBIGServiceConvenienceMethods' API method 'getHierarchyLevelPrev'
- Use the CodedNodeGraph API to resolve the immediate Associations of a given code, and check if they exist.

getHasChildrenbyCode

Either:

- Use the 'LexBIGServiceConvenienceMethods' API method 'getHierarchyLevelNext'
- Use the CodedNodeGraph API to resolve the immediate Associations of a given code, and check if they exist.

getIsRetiredbyCode

Use the 'LexBIGServiceConvenienceMethods' API method 'isCodeRetired'

getLocalNames

Use the 'LexBIGService' API method 'getSupportedCodingSchemes' - and extract the Names (local name, registered name, etc...) as needed.

Categories: VKC Contents | Documentation | LexEVS