

LexEVS 6.0 CTS2 Authoring 3 - Concept Domain Authoring Operation API

Contents of this Page

- [Introduction](#)
- [Interface](#)
- [Revision Information](#)
- [Authoring Functions](#)
 - [Create Operations](#)
 - [createConceptDomainCodeSystem](#)
 - [createConceptDomain](#)
 - [addConceptDomainProperty](#)
 - [addConceptDomainToValueSetBinding](#)
 - [Edit Operations](#)
 - [updateConceptDomainProperty](#)
 - [Remove Operations](#)
 - [removeConceptDomain](#)
 - [removeConceptDomainProperty](#)
 - [removeConceptDomainToValueSetBinding](#)
 - [Versionable Change Operations](#)
 - [updateConceptDomainStatus](#)
 - [activateConceptDomain](#)
 - [deactivateConceptDomain](#)
 - [updateConceptDomainVersionable](#)

CTS2 Links for LexEVS 6.0

- [CTS2 API Main Page](#)
- [Programmer's Guide Main Page](#)
- [LexEVS 6.0 Main Page](#)
- [LexEVS Current Release](#)

Introduction

LexEVS CTS2 Concept Domain Authoring API provides capability to author Concept Domain, maintain the binding between Concept Domain and the Value Sets and also the ability to create a Code System that can hold the Concept Domains.

Here are the authoring functions that can be performed on concept domain:

- **Create** - This function provides capability to create:
 - New Code System to hold concept domains
 - New Concept Domain
 - Add new property to Concept Domain
 - Concept Domain to Value Set binding
- **Edit** - This function provides capability to update:
 - Property of a Concept Domain
- **Remove** - This function provides capability to remove:
 - Concept Domain
 - Property of a Concept Domain
 - Concept Domain to Value Set binding
- **Versionable Change** - This function provides capability to modify versionable attributes of Concept Domain like Status, Effective Date, Expiration Date, isActive and Owner:
 - Concept Domain Status
 - Activate Concept Domain
 - De-Activate Concept Domain

Interface

`org.lexevs.cts2.author.ConceptDomainAuthoringOperation` is the main interface for all the authoring operations against Concept Domain. This interface can be accessed using main LexEVSCS2 interface, like:

```
org.lexevs.cts2.author.ConceptDomainAuthoringOperation cdAuthOp = new org.lexevs.cts2.LexEvsCTS2Impl().  
getAuthoringOperation().getConceptDomainAuthoringOperation();
```

Revision Information

All the authoring functions described here requires information about the author and revision/version id to be assigned to entities for each of these operations. These is done passing object **org.lexevs.cts2.core.update.RevisionInfo**. RevisionInfo object has following attributes:

- **java.lang.String changeAgent** - (Optional) The source that participated in this particular change.
- **java.lang.String changeInstruction** - (Optional) A human or machine readable set of instructions on how to apply this change.
- **java.lang.String revisionId** - (Mandatory) The unique identifier of this revision.
- **java.lang.Long editOrder** - (Optional) The relative order that this revision is to be applied if in a systemRelease.
- **java.util.Date revisionDate** - (Optional) The end date for which this version is operative (considered committed).
- **java.lang.String description** - (Optional) The description of the resource/change.
- **java.lang.String systemReleaseURI** - (Optional) The official URI of this release

Authoring Functions

Following sections contains detailed functions provided by ConceptDomainAuthoringOperation interface.

Create Operations

Create Operation provides capability to create a Code System to hold Concept Domains, Concept Domain, Property and binding between Concept Domain and Value Sets. For every entry that gets created, a unique revision(version) identifier will be assigned to that entry. This helps in retrieving snapshots of an entry based on its revision(version) identifier. This unique id can be passed in using the RevisionInfo object described above.

createConceptDomainCodeSystem

This function provides capability to create a new Code System to contain a set of Concept Domains. The Code System is created by defining the set of meta-data properties that describe it. At this point there is no concept domain added.

```
createConceptDomainCodeSystem(RevisionInfo revision, String codeSystemName, String codeSystemURI, String  
formalName, String defaultLanguage, long approxNumConcepts, String representsVersion, List<String> localNameList,  
List<org.LexGrid.commonTypes.Source> sourceList, Text copyright, Mappings mappings)
```

Description:	Creates new code system to hold concept domains.
Input:	• org.lexevs.cts2.core.update.RevisionInfo revision - (Mandatory) Contains revision information like unique RevisionId, change description, author information etc. • java.lang.String codingSchemeName - (Mandatory) Concept Domain Code System Name. • java.lang.String codingSchemeURI - (Mandatory) Concept Domain Code System URI. • java.lang.String formalName - (Optional) Formal name of a Concept Domain Code System. • java.lang.String defaultLanguage - (Optional) Default language of Concept Domain Code System. • long approxNumConcepts - (Optional) Approximate number of concept domains this Concept Domain Code System may contain. • java.lang.String representsVersion - (Mandatory) Initial version of the Concept Domain Code System. • java.util.List<java.lang.String> localNameList - (Optional) Any local name(s)/reference(s) for this Concept Domain Code System used within the Code System. • java.util.List<org.LexGrid.commonTypes.Source> sourceList - (Optional) Source(s) of this Concept Domain Code System. • org.LexGrid.commonTypes.Text copyright - (Optional) Information about rights held in and over the Concept Domain Code System. Typically, copyright information includes a statement about various property rights associated with the Concept Domain Code System, including intellectual property rights. • org.LexGrid.naming.Mappings mappings - (Mandatory) A list of all of the local identifiers and defining URI's that are used in the Concept Domain Code System.
Output:	org.LexGrid.codingSchemes.CodingScheme - Created Concept Domain Code System
Exception:	org.LexGrid.LexBIG.Exceptions.LBException

**Sample
Call:**

- *Step 1:* Instantiate ConceptDomainAuthoringOperation if it is not done yet:

```
org.lexevs.cts2.author.ConceptDomainAuthoringOperation cdAuthorOp = LexEvsCTS2Impl.  
defaultInstance().getAuthoringOperation().getConceptDomainAuthoringOperation();
```

- *Step 2:* Populate RevisionInfo object:

```
RevisionInfo revInfo = new RevisionInfo();  
revInfo.setChangeAgent("change Agent Name");  
revInfo.setChangeInstruction("here goes the change Instructions");  
revInfo.setDescription("description of the resource");  
revInfo.setEditOrder(1L);  
revInfo.setRevisionDate(new Date());  
revInfo.setRevisionId(UUID.randomUUID().toString());
```

- *Step 3:* Populate new concept domain code system meta data:

```
String codingSchemeURI = "urn:oid:11.11.22.999";  
String representsVersion = "1.0";  
  
String codingSchemeName = "Concept Domain Coding Scheme";  
String formalName = "CTS 2 API Created Concept Domain Code System";  
String defaultLanguage = "en";  
Long approxNumConcepts = new Long(1);  
List<String> localNameList = Arrays.asList();  
  
org.LexGrid.commonTypes.Source source = new org.LexGrid.commonTypes.Source();  
source.setContent("source");  
List<org.LexGrid.commonTypes.Source> sourceList = Arrays.asList(source);  
  
Text copyright = new Text();  
org.LexGrid.naming.Mappings mappings = new org.LexGrid.naming.Mappings();  
  
org.LexGrid.naming.SupportedLanguage supportedLang = new org.LexGrid.naming.SupportedLanguage();  
  
supportedLang.setLocalId("en");_  
supportedLang.setUri("URI_for_lang_en");_  
  
mappings.addSupportedLanguage(supportedLang);
```

- *Step 4:* call create method to create the concept domain code system:

```
CodingScheme codeScheme = cdAuthorOp.createConceptDomainCodeSystem(revInfo, codingSchemeName,  
codingSchemeURI, formalName, defaultLanguage, approxNumConcepts, representsVersion,  
localNameList, sourceList, copyright, mappings);
```

createConceptDomain

This function creates a concept domain to be included in a Code System. The new concept domain is defined by the set of meta-data properties that describe it.

```
createConceptDomain(String conceptDomainId, String conceptDomainName, String namespace, RevisionInfo  
revisionInfo, String description, String status, boolean isActive, Properties properties, String  
codeSystemNameOrURI, String codeSystemVersion)
```

Description:	Creates new concept domain in a code system.
---------------------	--

Input:	• <i>java.lang.String conceptDomainId</i> - (Mandatory) ID of a new concept domain. • <i>java.lang.String conceptDomainName</i> - (Mandatory) Name of a new concept domain. • <i>java.lang.String namespace</i> - (Mandatory) Namespace of a new concept domain. • <i>org.lexevs.cts2.core.update.RevisionInfo revision</i> - (Mandatory) Contains revision information like unique RevisionId, change description, author information etc. • <i>java.lang.String description</i> - (Mandatory) Description of a new concept domain. • <i>java.lang.String status</i> - (Optional) Status of new Concept Domain. • <i>boolean isActive</i> - (Optional) Status of new Concept Domain. • <i>org.LexGrid.commonTypes.Properties</i> - (Optional) List of properties for the new Concept Domain. • <i>java.lang.String codeSystemNameOrURI</i> - (Mandatory) Name or URI of a Code System that will hold this new Concept Domain. • <i>java.lang.String codeSystemVersion</i> - (Mandatory) Version of a Code System that will hold this new Concept Domain.
Output:	<i>java.lang.String</i> - concept domain id if created successfully
Exception:	<i>org.LexGrid.LexBIG.Exceptions.LBException</i>
Sample Call:	• Step 1: Instantiate ConceptDomainAuthoringOperation if it is not done yet: <pre>org.lexevs.cts2.author.ConceptDomainAuthoringOperation cdAuthorOp = LexEvsCTS2Impl. defaultInstance().getAuthoringOperation().getConceptDomainAuthoringOperation();</pre> • Step 2: Populate RevisionInfo object: <pre>RevisionInfo revInfo = new RevisionInfo(); revInfo.setChangeAgent("change Agent Name"); revInfo.setChangeInstruction("here goes the change Instructions"); revInfo.setDescription("description of the resource"); revInfo.setEditOrder(1L); revInfo.setRevisionDate(new Date()); revInfo.setRevisionId(UUID.randomUUID().toString());</pre> • Step 3: Populate a properties to be added: <pre>Property prop = new Property(); prop.setPropertyId("propertyId1"); prop.setPropertyName("propertyName"); prop.setIsActive(false); prop.setLanguage("en"); prop.setOwner("owner"); prop.setPropertyType(PropertyTypes.PROPERTY.name()); Text text = new Text(); text.setContent("content"); text.setDataType("Text datatype"); prop.setValue(text); Properties props = new Properties(); props.addProperty(prop);</pre> • Step 3: call create concept domain method by passing code system version and concept domain information: <pre>String conceptDomainId = cdAuthOp.createConceptDomain("CD00A", "Automobiles-CD", "Automobiles", revInfo, "Concept Domain for automobiles", "pending", false, props, "Automobile", "1.0",);</pre>

addConceptDomainProperty

This function provides capability to add a new property to a concept domain.

```
addConceptDomainProperty(String conceptDomainId, String namespace, Property newProperty, String  
codeSystemNameOrURI, String codeSystemVersion, RevisionInfo revisionInfo)
```

Description:	Add new property for a concept domain.
---------------------	--

Input:	• <i>java.lang.String conceptDomainId</i> - (Mandatory) Identifier of a Concept Domain to which a new property will be added. • <i>java.lang.String namespace</i> - (Mandatory) Namespace of a Concept Domain to which a new property will be added. • <i>org.LexGrid.commonTypes.Property newProperty</i> - (Mandatory) New property that will be added to the concept domain. • <i>java.lang.String codeSystemNameOrURI</i> - (Mandatory) Name or URI of a Code System that contains the concept domain. • <i>java.lang.String codeSystemVersion</i> - (Mandatory) Version of a Code System that contains the concept domain. • <i>org.lexvs.cts2.core.update.RevisionInfo revision</i> - (Mandatory) Contains revision information like unique RevisionId, change description, author information etc.
Output:	<i>boolean</i> - True; if addition of new property was success
Exception:	<i>org.LexGrid.LexBIG.Exceptions.LBException</i>
Sample Call:	• <i>Step 1:</i> Instantiate ConceptDomainAuthoringOperation if it is not done yet: <pre>org.lexvs.cts2.author.ConceptDomainAuthoringOperation cdAuthorOp = LexEvsCTS2Impl.defaultInstance().getAuthoringOperation().getConceptDomainAuthoringOperation();</pre> • <i>Step 2:</i> Populate RevisionInfo object: <pre>RevisionInfo revInfo = new RevisionInfo(); revInfo.setChangeAgent("change Agent Name"); revInfo.setChangeInstruction("here goes the change Instructions"); revInfo.setDescription("description of the resource"); revInfo.setEditOrder(1L); revInfo.setRevisionDate(new Date()); revInfo.setRevisionId(UUID.randomUUID().toString());</pre> • <i>Step 3:</i> Populate a property to be added: <pre>Property prop = new Property(); prop.setPropertyId("propertyId1"); prop.setPropertyName("propertyName"); prop.setIsActive(false); prop.setLanguage("en"); prop.setOwner("owner"); prop.setPropertyType(PropertyTypes.PROPERTY.name()); Text text = new Text(); text.setContent("content"); text.setDataTypes("Text datatype"); prop.setValue(text);</pre> • <i>Step 4:</i> call add property method by passing the code system information, concept domain information and a new property: <pre>boolean added = cdAuthOp.addNewConceptDomainProperty("CD00A", "Automobiles-CD", prop, "Automobiles", "1.0", revInfo);</pre>

addConceptDomainToValueSetBinding

This function provides capability to create a value set to concept domain binding.

addConceptDomainToValueSetBinding(String conceptDomainId, String namespace, String codeSystemNameOrURI, String codeSystemVersion, URI valueSetURI, RevisionInfo revisionInfo)

Description:	Create a value set to concept domain binding.
Input:	• <i>java.lang.String conceptDomainId</i> - (Mandatory) Identifier of a Concept Domain. • <i>java.lang.String namespace</i> - (Mandatory) Namespace of a Concept Domain. • <i>java.lang.String codeSystemNameOrURI</i> - (Mandatory) Name or URI of a Code System that contains the concept domain. • <i>java.lang.String codeSystemVersion</i> - (Mandatory) Version of a Code System that contains the concept domain. • <i>java.net.URI valueSetURI</i> - (Mandatory) URI of a value set that will be binded to the concept domain. • <i>org.lexvs.cts2.core.update.RevisionInfo revision</i> - (Mandatory) Contains revision information like unique RevisionId, change description, author information etc.

Output:	boolean - True; if binding was success
Exception:	org.LexGrid.LexBIG.Exceptions.LBException
Sample Call:	- Step 1: Instantiate ConceptDomainAuthoringOperation if it is not done yet: <pre>org.lexevs.cts2.author.ConceptDomainAuthoringOperation cdAuthOp = LexEvsCTS2Impl.defaultInstance().getAuthoringOperation().getConceptDomainAuthoringOperation();</pre> - Step 2: Populate RevisionInfo object: <pre>RevisionInfo revInfo = new RevisionInfo(); revInfo.setChangeAgent("change Agent Name"); revInfo.setChangeInstruction("here goes the change Instructions"); revInfo.setDescription("description of the resource"); revInfo.setEditOrder(1L); revInfo.setRevisionDate(new Date()); revInfo.setRevisionId(UUID.randomUUID().toString());</pre> - Step 3: call add binding method by passing the code system information, concept domain information and value set uri: <pre>boolean added = cdAuthOp.addConceptDomainToValueSetBinding("CD00A", "Automobiles-CD", "Automobiles", "1.0", "some_value_set_uri", revInfo);</pre>

Edit Operations

The edit operation provides the capability to modify properties of a concept domain. For every entry that gets modified, a unique revision(version) identifier will be assigned to that entry. This helps in retrieving snapshots of an entry based on its revision(version) identifier. This unique id can be passed in using the RevisionInfo object described above.

updateConceptDomainProperty

This function provides capability to modify existing property of a concept domain.

```
updateConceptDomainProperty(String conceptDomainId, String namespace, Property changedProperty, String codeSystemNameOrURI, String codeSystemVersion, RevisionInfo revisionInfo)
```

Description:	Modifies existing property of a concept domain.
Input:	java.lang.String conceptDomainId - (Mandatory) Identifier of a Concept Domain that contains the property. java.lang.String namespace - (Mandatory) Namespace of a concept domain that contains the property. org.LexGrid.commonTypes.Property property - (Mandatory) Modified property. java.lang.String codeSystemNameOrURI - (Mandatory) Name or URI of a Code System that contains the concept domain. java.lang.String codeSystemVersion - (Mandatory) Version of a Code System that contains the concept domain. org.lexevs.cts2.core.update.RevisionInfo revision - (Mandatory) Contains revision information like unique RevisionId, change description, author information etc.
Output:	boolean - True; if update was success
Exception:	org.LexGrid.LexBIG.Exceptions.LBException

Sample Call:	- Step 1: Instantiate ConceptDomainAuthoringOperation if it is not done yet: <pre>org.lexevs.cts2.author.ConceptDomainAuthoringOperation cdAuthorOp = LexEvsCTS2Impl.defaultInstance().getAuthoringOperation().getConceptDomainAuthoringOperation();</pre> - Step 2: Populate RevisionInfo object: <pre>RevisionInfo revInfo = new RevisionInfo(); revInfo.setChangeAgent("change Agent Name"); revInfo.setChangeInstruction("here goes the change Instructions"); revInfo.setDescription("description of the resource"); revInfo.setEditOrder(1L); revInfo.setRevisionDate(new Date()); revInfo.setRevisionId(UUID.randomUUID().toString());</pre> - Step 3: Populate a modified property. Just modified isActive to 'true' and language to 'eng' in property 'propertyId1': <pre>Property prop = new Property(); prop.setPropertyId("propertyId1"); prop.setPropertyName("propertyName"); prop.setIsActive(true); prop.setLanguage("eng");</pre> - Step 4: call update property method by passing the code system , concept domain information and the modified property: <pre>boolean updated = cdAuthOp.updateConceptDomainProperty("CD0001", "Automobiles-CD", prop, "Automobiles", "1.0", revInfo);</pre>
--------------	---

Remove Operations

Remove operation provides capability to remove concept domain, its property and its binding to value set.

removeConceptDomain

This function provides capability to remove a concept domain.

```
removeConceptDomain(String conceptDomainId, String namespace, String codeSystemNameOrURI, String codeSystemVersion, RevisionInfo revisionInfo)
```

Description:	Removes a concept domain from code system.
Input:	java.lang.String conceptDomainId - (Mandatory) Identifier of a Concept Domain to be removed. java.lang.String namespace - (Mandatory) Namespace of a Concept Domain to be removed. java.lang.String codeSystemNameOrURI - (Mandatory) Name or URI of a Code System that contains the concept domain to be removed. java.lang.String codeSystemVersion - (Mandatory) Version of a Code System that contains the concept domain to be removed. org.lexevs.cts2.core.update.RevisionInfo revision - (Mandatory) Contains revision information like unique RevisionId, change description, author information etc.
Output:	boolean - True; if remove was success
Exception:	org.LexGrid.LexBIG.Exceptions.LBException

Sample Call:	- Step 1: Instantiate ConceptDomainAuthoringOperation if it is not done yet: <pre>org.lexevs.cts2.author.ConceptDomainAuthoringOperation cdAuthorOp = LexEvsCTS2Impl.defaultInstance().getAuthoringOperation().getConceptDomainAuthoringOperation();</pre> - Step 2: Populate RevisionInfo object: <pre>RevisionInfo revInfo = new RevisionInfo(); revInfo.setChangeAgent("change Agent Name"); revInfo.setChangeInstruction("here goes the change Instructions"); revInfo.setDescription("description of the resource"); revInfo.setEditOrder(1L);revInfo.setRevisionDate(new Date()); revInfo.setRevisionId(UUID.randomUUID().toString());</pre> - Step 3: call remove concept domain method by passing the code system and concept domain information: <pre>boolean removed = cdAuthOp.deleteConcept("CD0001", "Automobiles-CD", "urn:oid:11.11.0.99", "1.0", revInfo);</pre>
---------------------	---

removeConceptDomainProperty

This function provides capability to remove property of a concept domain.

```
removeConceptDomainProperty(String conceptDomainId, String namespace, Property property, String codeSystemNameOrURI, String codeSystemVersion, RevisionInfo revisionInfo)
```

Description:	Removes a property of a concept domain.
Input:	java.lang.String conceptDomainId - (Mandatory) Concept Domain identifier that contains the property. java.lang.String namespace - (Mandatory) Namespace of a concept domain that contains the property. org.LexGrid.commonType.Property property - (Mandatory) Property that needs to be removed. java.lang.String codeSystemNameOrURI - (Mandatory) Code System URI/name that contains the concept domain. java.lang.String representsVersion - (Mandatory) Version of the Code System that contains the concept domain. org.lexevs.cts2.core.update.RevisionInfo revision - (Mandatory) Contains revision information like unique RevisionId, change description, author information etc.
Output:	boolean - True; if remove was success
Exception:	org.LexGrid.LexBIG.Exceptions.LBException

Sample Call:	- Step 1: Instantiate ConceptDomainAuthoringOperation if it is not done yet: <pre>org.lexevs.cts2.author.ConceptDomainAuthoringOperation cdAuthorOp = LexEvsCTS2Impl.defaultInstance().getAuthoringOperation().getConceptDomainAuthoringOperation();</pre> - Step 2: Populate RevisionInfo object: <pre>RevisionInfo revInfo = new RevisionInfo(); revInfo.setChangeAgent("change Agent Name"); revInfo.setChangeInstruction("here goes the change Instructions"); revInfo.setDescription("description of the resource"); revInfo.setEditOrder(1L); revInfo.setRevisionDate(new Date()); revInfo.setRevisionId(UUID.randomUUID().toString());</pre> - Step 3: Populate property that needs to be removed: <pre>Property propertyToRemove = new Property(); propertyToRemove.setPropertyId("p1");</pre> - Step 4: Call remove property method by passing the code system, concept domain and property information: <pre>boolean removed = cdAuthOp.removeConceptDomainProperty("CD0001", "Automobiles-CD", propertyToRemove, "urn:oid:11.11.0.99", "1.0", revInfo);</pre>
---------------------	--

removeConceptDomainToValueSetBinding

This function provides capability to remove value set to concept domain binding.

removeConceptDomainToValueSetBinding(String conceptDomainId, String namespace, String codeSystemNameOrURI, String codeSystemVersion, URI valueSetURI, RevisionInfo revisionInfo)

Description:	Removes value set to concept domain binding.
Input:	java.lang.String conceptDomainId - (Mandatory) Concept Domain identifier. java.lang.String namespace - (Mandatory) Namespace of a concept domain java.lang.String codeSystemNameOrURI - (Mandatory) Code System URI/name that contains the concept domain. java.lang.String representsVersion - (Mandatory) Version of the Code System that contains the concept domain. java.net.URI valueSetURI - (Mandatory) URI of a value set. org.lexevs.cts2.core.update.RevisionInfo revision - (Mandatory) Contains revision information like unique RevisionId, change description, author information etc.
Output:	boolean - True; if binding remove was success
Exception:	org.LexGrid.LexBIG.Exceptions.LBException

Sample Call:	- Step 1: Instantiate ConceptDomainAuthoringOperation if it is not done yet: <pre>org.lexevs.cts2.author.ConceptDomainAuthoringOperation cdAuthorOp = LexEvsCTS2Impl.defaultInstance().getAuthoringOperation().getConceptDomainAuthoringOperation();</pre> - Step 2: Populate RevisionInfo object: <pre>revisionInfo revInfo = new RevisionInfo(); revInfo.setChangeAgent("change Agent Name"); revInfo.setChangeInstruction("here goes the change Instructions"); revInfo.setDescription("description of the resource"); revInfo.setEditOrder(1L); revInfo.setRevisionDate(new Date()); revInfo.setRevisionId(UUID.randomUUID().toString());</pre> - Step 3: Call remove binding method by passing the code system, concept domain and value set information: <pre>boolean removed = cdAuthOp.removeConceptDomainToValueSetBinding("CD0001", "Automobiles-CD", "urn:oid:11.11.0.99", "1.0", "some_value_set_uri", revInfo);</pre>
---------------------	---

Versionable Change Operations

Versionable Change operation provides capability to modify versionable attributes of Concept Domain like Status, Effective Date, Expiration Date, isActive and Owner.

updateConceptDomainStatus

This function modifies the status of a concept domain.

updateConceptDomainStatus(String conceptDomainId, String namespace, String newStatus, String codeSystemNameOrURI, String codeSystemVersion, RevisionInfo revisionInfo)

Description:	Modifies the status of a concept domain.
Input:	java.lang.String conceptDomainId - (Mandatory) identifier of a concept domain. java.lang.String namespace - (Mandatory) Mamespace of a concept domain. java.lang.String status - (Mandatory) Modified status value. java.lang.String codeSystemNameOrURI - (Mandatory) Code System URI that contains the concept domain. java.lang.String codeSystemVersion - (Mandatory) Version of the Code System that contains the concept domain. org.lexevs.cts2.core.update.RevisionInfo revision - (Mandatory) Contains revision information like unique RevisionId, change description, author information etc.
Output:	boolean - True; if update was success
Exception:	org.LexGrid.LexBIG.Exceptions.LBException

Sample Call:	• <i>Step 1:</i> Instantiate ConceptDomainAuthoringOperation if it is not done yet: <pre>org.lexevs.cts2.author.ConceptDomainAuthoringOperation cdAuthorOp = LexEvsCTS2Impl.defaultInstance().getAuthoringOperation().getConceptDomainAuthoringOperation();</pre> • <i>Step 2:</i> Populate RevisionInfo object: <pre>revisionInfo revInfo = new RevisionInfo(); revInfo.setChangeAgent("change Agent Name"); revInfo.setChangeInstruction("here goes the change Instructions"); revInfo.setDescription("description of the resource"); revInfo.setEditOrder(1L); revInfo.setRevisionDate(new Date()); revInfo.setRevisionId(UUID.randomUUID().toString());</pre> • <i>Step 2:</i> Call status change method by passing the code system, concept domain and status information: <pre>boolean changed = cdAuthOp.updateConceptDomainStatus("CD000001", "Automobiles-CD", "Active", "urn:oid:11.11.0.99", "1.0", revInfo);</pre>
---------------------	---

activateConceptDomain

This function activates concept domain so that it can be accessed in the terminology service.

activateConceptDomain(String conceptDomainId, String namespace, String codeSystemNameOrURI, String codeSystemVersion, RevisionInfo revisionInfo)

Description:	Activates concept domain.
Input:	• <i>java.lang.String conceptDomainId</i> - (Mandatory) identifier of a concept domain. • <i>java.lang.String namespace</i> - (Mandatory) Mamespace of a concept domain. • <i>java.lang.String codeSystemNameOrURI</i> - (Mandatory) Code System URI that contains the concept domain. • <i>java.lang.String codeSystemVersion</i> - (Mandatory) Version of the Code System that contains the concept domain. • <i>org.lexevs.cts2.core.update.RevisionInfo revision</i> - (Mandatory) Contains revision information like unique RevisionId, change description, author information etc.
Output:	<i>boolean</i> - True; if activation was success
Exception:	<i>org.LexGrid.LexBIG.Exceptions.LBException</i>

Sample Call:	- Step 1: Instantiate ConceptDomainAuthoringOperation if it is not done yet: <pre>org.lexevs.cts2.author.ConceptDomainAuthoringOperation cdAuthorOp = LexEvsCTS2Impl.defaultInstance().getAuthoringOperation().getConceptDomainAuthoringOperation();</pre> - Step 2: Populate RevisionInfo object: <pre>revisionInfo revInfo = new RevisionInfo(); revInfo.setChangeAgent("change Agent Name"); revInfo.setChangeInstruction("here goes the change Instructions"); revInfo.setDescription("description of the resource"); revInfo.setEditOrder(1L); revInfo.setRevisionDate(new Date()); revInfo.setRevisionId(UUID.randomUUID().toString());</pre> - Step 2: Call activate method by passing the code system and concept domain information: <pre>boolean activated = cdAuthOp.activateConceptDomain("CD000001", "Automobiles-CD", "urn:oid:11.11.0.99", "1.0", revInfo);</pre>
---------------------	---

deactivateConceptDomain

This function deactivates concept domain so that it can no longer be accessed in the terminology service.

```
deactivateConceptDomain(String conceptDomainId, String namespace, String codeSystemNameOrURI, String codeSystemVersion, RevisionInfo revisionInfo)
```

Description:	Deactivates concept domain.
Input:	java.lang.String conceptDomainId - (Mandatory) identifier of a concept domain. java.lang.String namespace - (Mandatory) Mamespace of a concept domain. java.lang.String codeSystemNameOrURI - (Mandatory) Code System URI that contains the concept domain. java.lang.String codeSystemVersion - (Mandatory) Version of the Code System that contains the concept domain. org.lexevs.cts2.core.update.RevisionInfo revision - (Mandatory) Contains revision information like unique RevisionId, change description, author information etc.
Output:	boolean - True; if deactivation was success
Exception:	org.LexGrid.LexBIG.Exceptions.LBException

Sample Call:	- Step 1: Instantiate ConceptDomainAuthoringOperation if it is not done yet: <pre>org.lexevs.cts2.author.ConceptDomainAuthoringOperation cdAuthorOp = LexEvsCTS2Impl.defaultInstance().getAuthoringOperation().getConceptDomainAuthoringOperation();</pre> - Step 2: Populate RevisionInfo object: <pre>revisionInfo revInfo = new RevisionInfo(); revInfo.setChangeAgent("change Agent Name"); revInfo.setChangeInstruction("here goes the change Instructions"); revInfo.setDescription("description of the resource"); revInfo.setEditOrder(1L); revInfo.setRevisionDate(new Date()); revInfo.setRevisionId(UUID.randomUUID().toString());</pre> - Step 2: Call deactivate method by passing the code system and concept domain information: <pre>boolean deactivated = cdAuthOp.deactivateConceptDomain("CD000001", "Automobiles-CD", "urn:oid:11.11.0.99", "1.0", revInfo);</pre>
--------------	--

updateConceptDomainVersionable

This function provides capability to modify concept domain versionable attributes like effective date, expiration date, owner, status etc.

updateConceptDomainVersionable(String conceptDomainId, String namespace, Versionable changedVersionable, String codeSystemNameOrURI, String codeSystemVersion, RevisionInfo revisionInfo)	
Description:	Update concept domain versionable attributes like effective date, expiration date, owner, status etc.
Input:	- java.lang.String conceptDomainId - (Mandatory) identifier of a concept domain. - java.lang.String namespace - (Mandatory) Mamespace of a concept domain. - org.LexGrid.commonTypes.Versionable changedVersionable - (Mandatory) versionable (like:owner, effectiveDate, expirationDate, status etc) changes. - java.lang.String codeSystemNameOrURI - (Mandatory) Code System URI that contains the concept domain. - java.lang.String codeSystemVersion - (Mandatory) Version of the Code System that contains the concept domain. - org.lexevs.cts2.core.update.RevisionInfo revision - (Mandatory) Contains revision information like unique RevisionId, change description, author information etc.
Output:	boolean - True; if update was success
Exception:	org.LexGrid.LexBIG.Exceptions.LBException

**Sample
Call:**

- *Step 1:* Instantiate ConceptDomainAuthoringOperation if it is not done yet:

```
org.lexevs.cts2.author.ConceptDomainAuthoringOperation cdAuthorOp = LexEvsCTS2Impl.  
defaultInstance().getAuthoringOperation().getConceptDomainAuthoringOperation();
```

- *Step 2:* Populate RevisionInfo object:

```
revisionInfo revInfo = new RevisionInfo();  
revInfo.setChangeAgent("change Agent Name");  
revInfo.setChangeInstruction("here goes the change Instructions");  
revInfo.setDescription("description of the resource");  
revInfo.setEditOrder(1L);  
revInfo.setRevisionDate(new Date());  
revInfo.setRevisionId(UUID.randomUUID().toString());
```

- *Step 2:* Populate versionable changes:

```
Versionable changedVersionable = new Versionable();  
changedVersionable.setEffectiveDate(new Date());  
changedVersionable.setIsActive(true);  
changedVersionable.setOwner("new Owner");  
changedVersionable.setStatus("new status");
```

- *Step 3:* Call change versionable method by passing the code system, concept domain and changed versionable information:

```
boolean changed = cdAuthOp.updateConceptDomainVersionable("CD000001", "Automobiles-CD",  
changedVersionable, "urn:oid:11.11.0.99", "1.0", revInfo);
```