LexEVS 5.x Local Runtime Administration

Contents of this Page

Introduction

This document is a section of the Administration Guide.

LexEVS Configuration Options

The LexEVS software, documentation, indexes, and system logs are located in the {LEXBIG_DIRECTORY} (e.g., /usr/local/packages/LexBIG or c:\lexbig). These files may be part of the local file system and may require backup procedures to meet servicability and recovery requirements for your organization.

LexEVS uses basic database indexes, but also includes a separate indexing facility using Apache Lucene. Lucene Index files are stored in a directory as specified in the `lbconfig.props` file `index_location` variable.

What Is a Coding Scheme Manifest?

A "Coding Scheme Manifest" (or simply "manifest" as used interchangeably in this document) allows the user to set values for a coding scheme while loading or converting a LexGrid "XML", "NCI MetaThesaurus", "NCI OWL","OWL", "OBO", "UMLS RRF File", or"HL7 RIM Database" source to LexGrid format.

What Is a Coding Scheme?

Coding Scheme is the term that is used to represent an ontology/terminology being loaded or converted. In the LexGrid data model a terminology is represented as a coding scheme and it can reference other coding schemes. An example of coding scheme is "Amino Acid" which is described in the "amino acid.owl" file.

A Coding Scheme has some meta information about it; values like 'formal name', 'local names', 'default language', 'version', 'copyright', 'sources' to name some.

Why Do We Need a Coding Scheme Manifest?

When a terminology is being converted to the LexGrid data model from its native format (in this case OWL), Coding Scheme information is read from the source file. Sometimes values may be missing (not provided or invalid) or the author/user of the terminology wants to override or set default values despite (or in addition to) what is provided in the source file. This can be accomplished using "manifest" files along with the source file.

How Do We Create a Coding Scheme Manifest File?

A coding scheme manifest file is a valid XML file, conforming to the schema defined by: http://LexGrid.org/schema/LexBIG/2007/01/CodingSchemeManifestList.xsd

This XML file can define values for one or more coding schemes you are dealing with. Some coding scheme meta-information may not easily map to information in the source file. In this case a manifest file is of great help to bridge the gap and control the information flow while mapping to the LexGrid model. A detailed model of the LexGrid Coding Scheme and its fields can be found online. Structure of the schema for the manifest file is explained in the following table (manifest components refer to the original LexGrid model schema namespaces and types):

- Coding Scheme Manifest entry field: **id**
  - Type: lgCommon:registeredName
  - Required: Yes
  - Override flag set: Not applicable
  - Description:
    The registered name is the key used to find a coding scheme (for example a unique URL or namespace by which other people access same coding scheme). This String value will be used to identify the manifest entry in the manifest file for the coding scheme too. For example the registered name for coding scheme "Amino-acid" is http://www.co-ode.org/ontologies/amino-acid/2006/05/18/amino-acid.owl#. This string is also set as the coding scheme's registered name field in the LexGrid model.

- Coding Scheme Manifest entry field: **codingScheme**

    - Type: lgBuiltin:localId
    - Required: No
    - Override flag set: Yes
    - Description:
      This value will be set for 'coding scheme name' in the LexGrid format counterpart. If the override flag is set to 'true', the value provided in the source file will be replaced with this one. Otherwise, this value is treated as a default value and used only if the value is not provided in the source file.

- Coding Scheme Manifest entry field: **entityDescription**
    - Type: lgCommon:entityDescription
    - Required: No
    - Override flag set: Yes
    - Description:
      This value will be set for 'coding scheme description' in the LexGrid format counterpart. If the override flag is set to 'true', the value provided in the source file will be replaced with this one. Otherwise, this value is treated as a default value and used only if the value is not provided in the source file.

- Coding Scheme Manifest entry field: **formalName**
    - Type: lgBuiltin:tsCaseIgnoreIA5String
    - Required: No
    - Override flag set: Yes
    - Description:
      This value will be set for 'coding scheme formal name' in the LexGrid format counterpart. If the override flag is set to 'true', the value provided in the source file will be replaced with this one. Otherwise, this value is treated as a default value and used only if the value is not provided in the source file.

- Coding Scheme Manifest entry field: **registeredName**
    - Type: lgCommon:registeredName
    - Required: No
    - Override flag set: Yes
    - Description:
      This value will be set for 'coding scheme registered name' in the LexGrid format counterpart. If the override flag is set to 'true', the value provided in the source file will be replaced with this one. Otherwise, this value is treated as a default value and used only if the value is not provided in the source file.

- Coding Scheme Manifest entry field: **defaultLanguage**
    - Type: lgCommon:defaultLanguage
    - Required: No
    - Override flag set: Yes
    - Description:
      This value will be set for 'coding scheme default language' in the LexGrid format counterpart. If the override flag is set to 'true', the value provided in the source file will be replaced with this one. Otherwise, this value is treated as a default value and used only if the value is not provided in the source file.

- Coding Scheme Manifest entry field: **representsVersion**
    - Type: lgCommon:version
    - Required: No
    - Override flag set: Yes
- Description:
  This value will be set for 'coding scheme version' in the LexGrid format counterpart. If the override flag is set to 'true', the value provided in the source file will be replaced with this one. Otherwise, this value is treated as a default value and used only if the value is not provided in the source file.

- Coding Scheme Manifest entry field: **localName**
    - Type: lgBuiltin:tsCaseIgnoreIA5String
    - Required: No
    - "To Add" flag set: Yes
    - Description:
      This value will be added for 'coding scheme local names'. If the add flag is set to 'true', this value will be added to the list of local names (if not there already). Otherwise, this value is treated as the default value and used only if the value is not provided in the source file.

- Coding Scheme Manifest entry field: **source**
    - Type: lgCommon:source
    - Required: No
    - "To Add" flag set: Yes
    - Description:
      This value will be added for 'coding scheme sources'. If the add flag is set to 'true', this value will be added to the list of sources (if not there already). Otherwise, this value is treated as the default value and used only if the value is not provided in the source file.

- Coding Scheme Manifest entry field: **copyright**
    - Type: lgCommon:text
    - Required: No
    - Override flag set: Yes
    - Description:
      This value will be set for 'coding scheme copyright' in the LexGrid format counterpart. If the override flag is set to 'true', the value provided in the source file will be replaced with this one. Otherwise, this value is treated as a default value and used only if the value is not provided in the source file.

- Coding Scheme Manifest entry field: **mappings**
  - Type: lgCS:mappings
  - Required: No
  - "To Add" flag set: Yes
  - Description:
    This value will be added for 'coding scheme mappings'. If the add flag is set to 'true', this value will be added to the list of mappings (if not there already). Otherwise, this value is treated as the default value and used only if the value is not provided in the source file.
- Coding Scheme Manifest entry field: **associationDefinitions**
  - Type: lgRel:association
  - Required: No
  - "To Add" flag set: Yes
  - Description:
    This value will be added for 'coding scheme associations'. If the add flag is set to 'true', this value will be added to the list of associations (if not there already). Otherwise, this value is treated as the default value and used only if the value is not provided in the source file.

Note

This option is used internally by the system to provide default recognition of some common associations. It is typically not necessary to provide this value, however, since association definitions are automatically derived from the source.

What Code Changes May Be Required To Use a Manifest File?

If you want to use the manifest file, you can supply the manifest file URI to the following methods when Loading NCI OWL or generic OWL Loads:

"org.LexGrid.LexBIG.Extensions.Load.OWL_Loader.load()"

"org.LexGrid.LexBIG.Extensions.Load.OWL_Loader.validate()"

An example code snipped:

Java Code Snippet

```
LexBIGService lbs = new LexBIGServiceImpl();
LexBIGServiceManager lbsm = lbs.getServiceManager(null);
OWL_Loader loader = (OWL_Loader) lbsm.getLoader("OWLLoader");

if (toValidateOnly)
{
    loader.validate(source, manifest, vl);
    System.out.println("VALIDATION SUCCESSFUL");
}
else
{
    loader.load(new File("resources/testData/amino-acid.owl").toURI(),
                new File("resources/testData/aa-manifest.xml").toURI(),true,true);
}
```

For all other manifest loads the following methods are employed.

{include:SupplyNonOwlFileUri Snippet}}

Database Configuration

Before You Begin

This section provides an overview of the components as related to system adminstration, backup, and recovery. Individual organizations may have their own backup and diaster recovery procedure. In every case users created for use by the LexEVS application must have read, write and delete access to the dbms instance.

Database systems as described in the section *Required Software—Not Included in LexEVS* provide the storage for vocabularies loaded into LexEVS. For each vocabulary version loaded into LexEVS a new database is created. As defined in the `lbconfig.props` files the `db_prefix` variable is used to create the database name.

For example with `db_prefix=lexbig`, each new vocabulary version that is loaded a new database is created using an incremental counter.

- lexbig1

- lexbig2
- lexbig3
- lexbigN

Depending on backup strategy, system administrators will need to be aware that multiple databases are being created and may need backup procedures to meet servicability and recovery requirements for your organization.

MySQL Configuration

MySQL properties

PostGreSQL Configuration

PostgreSQL properties