Using EMF Annotations to Drive Program Behavior

February 19, 2014

Innovative Solutions For Mission Critical Systems

Preview for:

eclipse CON 2014

Innovative Solutions For Mission Critical Systems

Summary:

This presentation uses the example EMF Library model to demonstrate how EAnnotations can be used to modify the behavior of an application.

The approach used by CohesionForce to identify model parameters to be used in a search context is also provided.

Experience Level: Intermediate

While the concepts discussed should be useful to all experience levels, some of the coding examples will require some level of familiarity in developing with EMF.

What are EMF Annotations?

According to the EMF Book:

“Annotations constitute a mechanism by which additional information can be attached to any object in an Ecore model.”

This presentation uses the term EAnnotation to refer to EMF Annotations. The term Annotation is used throughout the Eclipse community for many different purposes.

Common Uses of EAnnotations:

Add documentation to generated code:
  source: http://www.eclipse.org/emf/2002/GenModel
  key: documentation
  value: String used in documentation

Generate code that uses the Object Constraint Language:
  Very detailed description here:
  http://wiki.eclipse.org/OCL/OCLinEcore

ECore files created from XML Schema make prolific use of EAnnotations

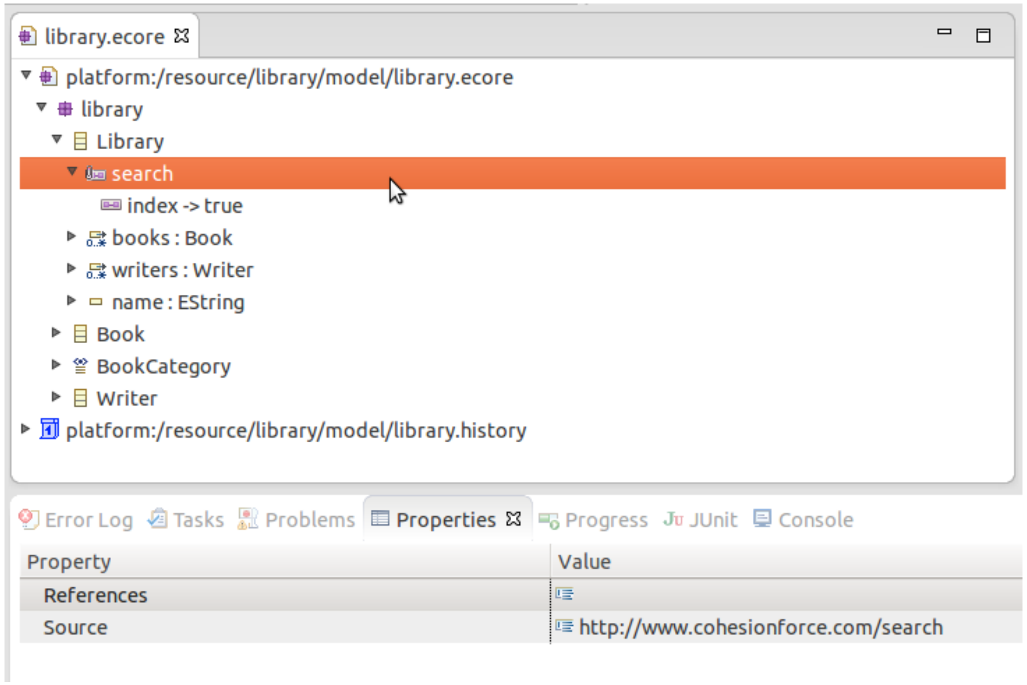
To add EAnnotations,

1. Right click on an element in the ecore editor and select “New Child->EAnnotation”.

2. Fill in the Source property.

3. Right click on the EAnnotation and select “New Child->Details Entry”

4. Fill in the key and value properties of the Entry.



library.ecore

```
▼ platform:/resource/library/model/library.ecore
  ▼ library
    ▼ Library
      ▼ search
          index -> true
      ▶ books : Book
      ▶ writers : Writer
      ▶ name : EString
    ▶ Book
    ▶ BookCategory
    ▶ Writer
▶ platform:/resource/library/model/library.history
```

Error Log | Tasks | Problems | Properties | Progress | JUnit | Console

Property	Value
References	
Source	http://www.cohesionforce.com/search

Innovative Solutions For Mission Critical Systems

Demonstrate of working with EAnnotations in the ECore Editor

When the code is generated, the annotations are added in the PackageImpl class. An example of the generated code is shown below:

```
protected void createSearchAnnotations() {
       String source = "http://www.cohesionforce.com/search";
       addAnnotation
         (libraryEClass,
          source,
          new String[] {
               "index", "true"
          });
}
```

Accessing EAnnotations in code:

Since the EAnnotation is an attribute of the base EModelElement class, it is accessible for pretty much everything in an EMF Object by moving from the concrete structure to the meta level.

From the concrete EObject, the EClass is available through the .eClass() method.

The data classes for most code generated from EMF models extends the EObject interface.

Below is an example of accessing an EAnnotation from the EClass of an EObject. The variable EMFIndexUtil.SOURCE is a static final String set to "http://www.cohesionforce.com/search", which is used for the source of the EAnnotation.

```
public void indexContents(EObject obj) {
        logger.debug("Indexing contents of {}", obj.eClass().getName());
        EAnnotation annotation = obj.eClass().getEAnnotation(
                EMFIndexUtil.SOURCE);

    .
    .
    .
```

The features of a class are available through the EClass of an EObject.

```
for (EStructuralFeature feature :
            obj.eClass().getEAllStructuralFeatures())
{
    // Do something with the feature
}
```

The following slide shows the method of retrieving EAnnotations from an EStructuralFeature.

```
protected void indexFeature(Document doc, EStructuralFeature feature,
                              EObject object) {

     boolean recurse = false;
     boolean tokenize = false;
     boolean index = false;

     EAnnotation annotation = feature.getEAnnotation(EMFIndexUtil.SOURCE);
     if (annotation != null) {
          if (annotation.getDetails().containsKey(EMFIndexUtil.RECURSE_KEY)) {
               recurse = true;
          }
          if (annotation.getDetails().containsKey(EMFIndexUtil.TOKENIZE_KEY)) {
               tokenize = true;
          }
          if (annotation.getDetails().containsKey(EMFIndexUtil.INDEX_KEY)) {
               index = true;
          }
     } else {
          // Bail out early if this feature should not be indexed
          return;
     }
     .
     .
     .
```

Any questions about creating EAnnotations, generating code, or accessing EAnnotations?

The following section describes a capability created by CohesionForce named EMFIndex that applies EMF Annotations to determine how data instances are indexed using Apache Lucene.

This capability will be released publicly under the Eclipse Public License as a part of our participation in EclipseCon2014.

EMF Index

EMF Index is a technique for using EMF Annotations to determine how an instance of an EMF model should be indexed using Apache Lucene.

Annotations

The annotations defined for EMF Index are as follows:

Source: http://www.cohesionforce.com/search

Details:

Key: tokenize	Value: N/A
Key: recurse	Value: N/A
Key: index	Value: N/A

The annotations can be applied in three different places:

1. EClass – The index detail entry is used to indicate that instances of this class should be indexed as a unique document with Apache Lucene. The unique document will always include the following terms:

URI of the containing Resource

FragmentURI in the containing Resource to the class instance

EClass name of the class instance

2. EAttribute – The tokenize detail entry is used for EString attributes to indicate that the value should be tokenized before indexing.

3. EReference (containment or non-containment) – The recurse detail entry is used to specify that the object referenced should also be folded into the same document as the class instance. This allows a hierarchical structure to be indexed in a flat document. An EReference marked with the index annotation will store the fragment URI of the referenced object.

To demonstrate the use of EAnnotations with Apache Lucene, we will start with a data set for the Library EMF Example. We did several searches with the Huntsville Madison County Library (http://hmcpl.org/) and converted the search results to instances of model data.

The dataset includes > 4500 books with lots of authors and many of the branches included as libraries.

We use a JUnit test framework to grab some timing metrics with both Lucene and also normal file loading for:

- Listing all Books
- Listing all Authors
- Listing all Books with the word “leader” in the title

Demo Time!

Observations on Lucene:

Indexing 9188 files took ~67 minutes. Most of this time seems to be spent on items marked with “tokenize=true”.

The file search version of the test simply uses the title.contains() method. This returns files with the word “leadership” in addition to the word “leader”.

There are many powerful search methods provided by Lucene that we have not yet exposed.

References:

EMF: Eclipse Modeling Framework, 2nd Edition
By Dave Steinberg, Frank Budinsky, Marcelo Paternostro, Ed Merks

EMF Examples:
http://www.eclipse.org/modeling/emf/downloads/?project=emf

EMF Source:
git://git.eclipse.org/gitroot/emf/org.eclipse.emf.git