Model Notification in EMF

Having learnt how to create, save and load an EMF model in the previous articles, the next topic to learn is how to listen to Model Changes thru Notifications.

In EMF, notifications are handled using EAdapters. There are 2 types of Notification. Object Notification or Content Notification.

Step 1 : Initialize your MODEL

```
AddressBook book = AddressbookFactory.eINSTANCE.createAddressBook();
Group group = AddressbookFactory.eINSTANCE.createGroup();
group.setGroupName("friends");
book.getGroup().add(group);

Contact contact1 = AddressbookFactory.eINSTANCE.createContact();
contact1.setContactName("ABC");
group.getContact().add(contact1);

Contact contact2 = AddressbookFactory.eINSTANCE.createContact();
contact2.setContactName("XYZ");
group.getContact().add(contact2);
```

Object Notification : Registering as an Object Listener

When you are interested in a particular Object 's attributes then we listen using Object Notification. This is achieved by registering as AdapterImpl .

```
//To listen whenever a particular Contact is changed.
contact1.eAdapters().add(new AdapterImpl(){
    @Override
    public void notifyChanged(Notification msg) {
        System.out.println("Contact Name Changed !");
        super.notifyChanged(msg);
    }
});

contact1.setContactName("NewABC");
contact2.setContactName("NewXYZ");
```

In the code above, we are registering as a listener to contact1 by adding ourselves as AdapterImpl to eAdapters list. AdapterImpl comes with an method named notifyChanged that would be invoked whenever model change happens.

Now whenever you change any attribute of contact1, notifyChanged would be invoked. Within notifyChanged we perform our action as desired.

But if you change the name of contact2, you will not be notified as what we just now achieved was Object Notification wherein we listen to a specific object alone.

Author : Annamalai C, Vikash Singh
Date : 24th October 2016

Content Notification : Registering as an Content Listener

Now as we saw in Object listener, we would have to register to every live object in the model. This isn't feasible. Therefore we register whats called as Content Listeners using EContentAdapter. In this case we need to register to the parent to be informed about its attributes along with its child references.

```
//To listen whenever a particular Contact is changed.
group.eAdapters().add(new EContentAdapter(){
    @Override
    public void notifyChanged(Notification msg) {
        System.out.println("Contact Name Changed !");
        super.notifyChanged(msg);
    }
});

contact1.setContactName("NewABC");
contact2.setContactName("NewXYZ");
```

Now I am interested in listening to whenever any contacts attribute is modified. In that case, I can become a EContentAdapter listener to a group which holds Contact or can become a listener to the AddressBook which contains all the groups and contacts.

Note : When using EContentAdapter, super.notifyChanged should be removed from the overridden notifyChanged method.

ABOUT ANCIT:

ANCIT Consulting is an Eclipse Consulting Firm located in the "Silicon Valley of Outsourcing", Bangalore. Offers professional Eclipse Support and Training for various Eclipse based Frameworks including RCP, EMF, GEF, GMF. Contact us on annamalai@ancitconsulting.com to learn more about our services.

Author : Annamalai C, Vikash Singh
Date : 24th October 2016

ANCIT CONSULTING