UniqueKey Example

by Steve Herrick, NeXT EOF Team

Overview

This example shows how to reserve a block of unique keys to support object insertion into an entity. The main window contains a master/detail relationship which supports update, insert, and delete. Adding employees requires the generation of a unique key for the new object. The UniqueKey class was written a resource to hand out the nextKey for each entity name. Instances of UniqueKey share a separate database channel to insure access for reserving new blocks of keys.

In this example, the master entity contains employee names and the detail displays equipment assigned to that employee. Each employee has a unique integer EmpId.

Program Organization

Explanation of the nib file

The File's Owner in the .nib file has been set to EOApplication. This was done by setting the File's Owner class in the inspector. EOApplication is necessary to implement autorelease pools under 3.2. This subclass of Application flushes the autorelease pool each cycle through the event queue. Special handling is used to make modal loops work correctly.

As mentioned in the release notes, the _main file was edited as follows:

```
#import <EOApplication.h>
```

was changed to

#import <eointerface/EOApplication.h>

There are three controllers in this example. The Employee controller has a master/detail relationship with its EquipmentForEmployee controller to display equipment assigned to a selected employee. An additional AddEquipmentForEmployee controller is used to allow selection of equipment not currently assigned to the employee. The MasterDetail object is the delegate of the Employee controller. It gets notification when inserts and delete operations are performed. This hook allows clean-up of equipment ownership when an employee leaves the company. MasterDetail has outlets for these controllers and supports actions from the release and assign buttons in the equipment detail.

Major Classes in the Application

MasterDetail	A coordination object which is the delegate of the Employee, EquipmentForEmployee, and AddEquipmentForEmployee controllers. It creates a UniqueKey instances to vend keys for the addition of employees.
UniqueKey	Reserves blocks of unique integer keys for a given entity name. UniqueKey uses a separate channel to insure access to its own unique_key entity. This entity (table) is used to store the maxKey for a given entityName. An instance of UniqueKey reserves a block of keys by incrementing maxKey by the block size. Optimistic locking is used for the update, so a re-try consists of a call to re-fetch the current data, increment maxKey, and attempt another update.

Other Peculiarities
This example uses the additional shared libraries libEOInterface_s.a, libEOAccess_s.a, and libFoundation_s.a. Also, Makefile.preamble uses:

OTHER_LDFLAGS = -all_load

to force the linking of shared library classes that are referenced at runtime by EOPalette.palette objects.

DB Scripts

installPEOPLE.sqlsybase installPEOPLE.sqloracle	Used to create and restore the databases for the Sybase and Oracle versions of this example.