EOF

Title: Updating Master Object Immediately on Insert Or Delete in Detail Controller

Entry Number:
Last Updated: 19 Apr 1995
Document Revision:

Keywords: EOF, insert, delete, detail

Questions

Q: How can I make inserts and deletes in a detail controller be reflected immediately in the master EO?

Answers

A: Consider a one-to-many Master/Detail scenario such as Publishers ->> Titles. The Titles controller maintains an array of Enterprise Objects (EOs) associated with the selected Publisher object. The selected Publisher object also keeps it's own array of EOs -- the value of it's toTitles relationship. In other words, there are two arrays of the same Titles, one in the detail EOController, and another in the master EO.

For master/detail relationships in EOF 1.0 and 1.1, insertions or deletions in one place are not automatically reflected in the other until a **saveToDataSource** is performed. That is, inserting a new Title into the detail controller will not cause a new object to automatically be inserted in the master EO's toTitles relationship, nor vice versa, until you save the changes to the data source. The same is true of deletions. If you want to keep these in sync before the save, you need to maintain this yourself.

There are two cases to consider. The most common is inserting into or deleting from the detail controller, and needing the change to be reflected in the master EO. This is easily accomplished with an EOController delegate that responds to the **controller:didInsertObject:**, **controller:didDeleteObject:**, and

controllerWillDiscardOperations: delegate methods for the detail controller, like this:

```
- (void)controller:(EOController *)controller didInsertObject:object
{
    NSMutableDictionary *dict = [NSMutableDictionary dictionary];
    EODetailDatabaseDataSource *dataSource = [controller dataSource];

    [dict setObject:[controller allObjects]
        forKey:[dataSource detailKey]];
    [[dataSource masterObject] takeValuesFromDictionary:dict]
}

- (void)controller:(EOController *)controller didDeleteObject:object
{
    NSMutableDictionary *dict = [NSMutableDictionary dictionary];
    EODetailDatabaseDataSource *dataSource = [controller dataSource];

    [dict setObject:[controller allObjects]
        forKey:[dataSource detailKey]];
    [[dataSource masterObject] takeValuesFromDictionary:dict]
}

- (BOOL)controllerWillDiscardOperations:(EOController *)controller
{
    EODatabaseDataSource  *source = [controller dataSource];
    if([source isKindOfClass:[EODetailDatabaseDataSource class]])
        return NO;

    return YES;
}
```

The **controllerWillDiscardOperations:** is necessary because otherwise, the next time the master redisplays, the EOQualifiedAssociation to the detail will check its cached pointer of the value it saw in the masterEO and when it notices the change it will tell the detail controller to refetch. Of course the detail has an enqueued insert or delete operation, so the request to fetch will cause it to ask if it should discard. The answer is no. You know that the detail controller array is actually in sync with the master and there's no need to refetch. This code will have the side affect of supressing the alert

asking whether to allow discard when the selection changed in the master. If you need to allow the user to change selection in the master controller when outstanding edits exist in a detail, you must discard or save them explicitly before changing the selection.

The other case is inserting into or deleting from the master EO's relationship array and needing the change to be reflected in the detail controller. To accomplish this, you can send a **contentsDidChange** message to the EOQualifiedAssociation for the detail controller. This can be put in a category on EOController, like this:

```
@interface EOController (Extensions)

(void)redisplayDetailControllerWithKey:(NSString *)key;

@end

@implementation EOController (Extensions)

(void)redisplayDetailControllerWithKey:(NSString *)key
{
    NSEnumerator *assocEnum = [[self associations] objectEnumerator];
    id assoc;

    while((assoc = [assocEnum nextObject])) {
        if([[assoc key] isEqual:key]) {
            [assoc contentsDidChange];
            /* Don't break, might have more than one controller for this key */
        }
    }
}

@end
```

This code would be called like this:

```
    ...
    /*
     * Publishers is the master, Titles is the detail.  After inserting
```

```
     * a new Title into the Publisher's relationship array, we need to
     * reflect the change in the detail controller.
     *
     * Assume selectedPublisher is the selected EO in the Publisher controller,
     * and newTitle is the new Title object that we want to insert.
     */
    oldTitles = [selectedPublisher toTitles];
    newTitles = [[[NSMutableArray alloc] initWithArray:oldTitles] autorelease];

    [newTitles addObject:newTitle]
    [selectedPublisher setToTitles:newTitles];

    /* Update the detail controller for the key "toTitles" */
    [titleController redisplayDetailControllerWithKey:@"toTitles"];

    ...
```

Valid for: EOF 1.0, EOF 1.1, NEXTSTEP 3.2 Developer, NEXTSTEP 3.3 Developer