associations
disableAssociationNotification
keys
reenableAssociationNotification
removeAssociation:

Getting the objects allObjects

Fetching objects fetch

Managing the selection setSelectionIndexes:
selectionIndexes
selectedObjects
clearSelection
selectNext
selectPrevious
setSelectsFirstObjectAfterFetch:
selectsFirstObjectAfterFetch

Editing objects associationDidEdit:
setValues:forObject:
endEditing

Sorting objects resort
setSortOrdering:
sortOrdering

Performing operations on objects
deleteObjectAtIndex:
deleteSelection
insertObjectAtIndex:
insertObject:atIndex:

Discarding changes discardEdits
discardOperations
isDiscardAllowed

Saving edits to objects saveToObjects
hasChangesForObjects
setSavesToObjectsAutomatically:
savesToObjectsAutomatically

Saving to the data source saveToDataSource
hasChangesForDataSource
setSavesToDataSourceAutomatically:
savesToDataSourceAutomatically

Controlling undo hasUndos
setUndoEnabled:
isUndoEnabled
markUndo
undo
releaseUndos
setMarksEveryOperation:
marksEveryOperation
setMaximumUndoMarks:
maximumUndoMarks

Redisplaying the user interface redisplay

Chaining controllers setNextController:
nextController

markUndo:
saveToDataSource:
saveToObjects:
selectNext:
selectPrevious:
undo:

(void)addAssociation:(EOAssociation *)anAssociation

Adds anAssociation to the set managed by the controller.

removeAssociation:, associations

(NSArray *)allObjects

Returns all enterprise objects accessible through the controller.

selectedObjects

(void)associationDidEdit:(EOAssociation *)anAssociation

Informs the controller that anAssociation has a pending edit. The controller gets anAssociation's k
the new value to its edit buffer.

If the controller is configured to save changes automatically then the changes are immediately save
controller doesn't save changes automatically, the changes don't take effect until the controller rec
message.

This method invokes the delegate method controller:association:didEditObject:key:value:.

key (EOAssociation), value (EOAssociation), savesToObjectsAutomatically, savesToDataSourc

(NSArray *)associations

Returns the controller's associations.

(BOOL)clearSelection

Empties the controller's selection. This method is a shortcut for sending setSelection: with an emp
redisplay the user interface objects. Returns YES if successful, NO if not.

redisplay

delete:sender

This action method invokes deleteSelection to delete all selected objects. Returns self if deleteSele
if it returns NO.

(BOOL)deleteObjectAtIndex:(unsigned int)anIndex

Deletes the object at anIndex, pending approval by the delegate. Returns YES if the object is actua
otherwise.

If the data source can't delete, this method opens an attention panel notifying the user and returns N
can delete, invokes endEditing and notifies the delegate with controller:willDelete:Object:. If the d
deletes the object and notifies the delegate with controller:didDeleteObject: and the controller's ass
contentsDidChange and selectionDidChange.

Deletion is an operation applied immediately to the object you don't need to send saveToObjects. I
configured to save to the data source automatically then the changes are immediately made to the d
the controller doesn't save changes automatically, the changes don't take effect in the data source u
receives a saveToDataSource message.

delete:, deleteSelection, savesToDataSourceAutomatically

(BOOL)deleteSelection

Deletes all objects in the selection in the manner described for deleteObjectAtIndex: (including inv
methods). Returns YES if all objects in the selection were successfully deleted (or if there were no
otherwise. If this method returns NO it's possible that only some objects in the selection have been

Deleting the selection is considered a single operation for purposes of undo, even if the controller r

delete:

(void)disableAssociationNotification

Allows you to specify that EOAssociationNotification messages should not be sent (they are sent b
disableAssociationNotification, you should send a corresponding reenableAssociationNotification.

reenableAssociationNotification

(void)discardEdits

Clears all edits pending in the controller, and also in user interface objects (by sending discardEdit
controller's associations). This method is only useful if the controller isn't configured to save to ob

savesToObjectsAutomatically

automatically.

savesToDataSourceAutomatically

discardOperations:sender

This action method invokes discardOperations and returns self.

(void)endEditing

Sends endEditing to each of the controller's associations. This method is invoked whenever the co
an end to editing in the user interface objects, which is whenever:

·The controller fetches.
·The controller saves changes to its objects or to its data source.
·An object is inserted or deleted.
·The selection changes.
·The undo stack is altered by a markUndo, undo, or releaseUndos message.

endEditing (EOAssociationNotification protocol)

(BOOL)fetch

Fetches objects from the controller's data source, returning YES if successful, NO if not.

The controller first invokes endEditing. Next, it sends controllerWillFetch: to its delegate, abortin
NO if the delegate returns NO. If the delegate returns YES, the controller then confirms that pendi
discarded by invoking isDiscardAllowed (it also confirms that any detail or linked controllers allo
be discarded). The controller aborts the fetch and returns NO if isDiscardAllowed returns NO or if
linked controllers don't allow discard.

After confirming that changes can be discarded, the controller discards all edits and operations, an
The controller then sends fetchObjects to its data source, and resynchronizes its selection with the

After fetching objects, the controller sends controllerDidFetchObjects: to its delegate and contents
associations. If the controller's selection changed as a result of fetching, also sends selectionDidCh
associations. Finally, the controller propagates the fetch message to all detail and linked controller

discardEdits, releaseUndos, selectsFirstObjectAfterFetch

fetch:sender

This action method invokes fetch and returns self if fetch returns YES, nil if it returns NO.

(BOOL)hasChangesForDataSource

Returns YES if there are any operations buffered for the data source, NO otherwise.

savesToDataSourceAutomatically, hasChangesForObjects, saveToObjects

(BOOL)hasUndos

Returns YES if the controller has any entries on its undo stack.

initWithDataSource:(id <EODataSources>)aDataSource

Initializes a newly allocated EOController to get its data from objects provided by aDataSource. Th
initializer for the EOController class. Returns self.

insert:sender

This action method uses insertObjectAtIndex: to insert a new object after the selected object, or if
the end of the array. Returns self, or nil if insertObjectAtIndex: returns nil.

insertObjectAtIndex:(unsigned int)anIndex

Inserts and selects a new object, created by the controller's data source, at anIndex by invoking ins
Invokes endEditing before performing the insertion. Returns the object that was inserted, or nil on

createObject (EODataSources protocol of the access layer)

insertObject:anEO atIndex:(unsigned int)anIndex

Inserts anEO at anIndex in the array of enterprise objects managed by the controller. Invokes endE
performing the insertion. Returns the object that was inserted, or nil on failure. Raises NSRangeEx
indicates a position past the end of the controller's array of objects.

Insertion is an operation applied immediately to the object you don't need to send saveToObjects.
configured to save to the data source automatically then the changes are immediately made to the d
the controller doesn't save changes automatically, the changes don't take effect in the data source u
receives a saveToDataSource message.

insert:, savesToDataSourceAutomatically

(BOOL)isDiscardAllowed

Verifies that discarding of edits not saved to objects and updates not saved to the data source is all

delegate. Again, if the delegate doesn't implement this method the controller opens an attention pa
save, cancel, or allow discard of operations if the user chooses to save operations or to cancel, this
Finally, if all of these conditions are passed, this method returns YES.

discardEdits, saveToObjects, saveToDataSource

(BOOL)isUndoEnabled

Returns YES if undo is enabled, NO otherwise. Undo is by default not enabled for controllers crea
though it is for controllers created in Interface Builder.

(NSArray *)keys

Returns the keys used by the controller's associations (which aren't necessarily all the keys used b
. These keys are the names of properties of the controller's enterprise objects, as used in the EOKe
protocol defined by the access layer. There may be duplicate keys in the returned array.

key (EOAssociation)

(void)markUndo

Ends editing and marks the current state of the undo stack as an undo point. An undo message reve
the last undo point marked. This method does nothing if undo isn't enabled or if a mark is already
stack.

isUndoEnabled, releaseUndos, endEditing

markUndo:sender

This action method invokes markUndo and returns self.

(BOOL)marksEveryOperation

Returns YES if an undo point is marked upon every insertion, deletion, and application of edits to
NO otherwise. Controllers created programmatically by default don't mark every operation, while
Interface Builder do.

deleteSelection

(unsigned int)maximumUndoMarks

Returns the maximum number of undo marks the controller will record. Marks added beyond this l
marks to fall off the queue. This method is useful for restricting memory usage. The default maxim
actually indicates no limit.

(void)redisplay

Sends contentsDidChange to all the controller's associations, causing them to redisplay their user i
needed. Also has the controller's detail controllers and next controller redisplay.

nextController

(void)reenableAssociationNotification

Allows you to specify that EOAssociationNotification messages should be sent (they are sent by d
disableAssociationNotification message. For each disableAssociationNotification, you should sen
reenableAssociationNotification. The final call to reenableAssociationNotification does not flush p
the associations [controller redisplay] must be explicitly called, if needed.

disableAssociationNotification

(void)releaseUndos

Clears all undo information. No undo is possible until a new undo point is established, either auton
markUndo.

marksEveryOperation

(void)removeAssociation:(EOAssociation *)anAssociation

Removes anAssociation from the set managed by the controller.

addAssociation:

(void)resort

forces the object array to be sorted and redisplayed. Note that the object array is automatically sort

setSortOrdering:

(BOOL)savesToDataSourceAutomatically

Returns YES if saving to objects, or inserting or deleting an object, results in the controller perform
saveToDataSource, NO if it doesn't. Controllers created programmatically by default don't save au
created in Interface Builder do.

saveToObjects, savesToObjectsAutomatically

(BOOL)savesToObjectsAutomatically

Ends editing and saves all pending operations for the controller itself, for its detail controllers, and
If none of the controllers has saved edits to their objects, this method has no effect use saveToObje
pending edits are applied to the objects before invoking saveToDataSource. Returns YES if all cha
saved to the data source, NO if any save operation fails or is refused by the delegate. If this metho
controller's data source doesn't support rolling back operations, some changes may have been save

In saving an object, the controller sends it a prepareForDataSource message (if it responds). If the
controller sends its delegate a controller:object:failedToPrepareForDataSource: message. If the del
EOContinueDataSourceFailureResponse the controller continues saving the remaining objects. If t
EORollbackDataSourceFailureResponse or doesn't implement the delegate method, the controller
operationsÐif the data source supports rollbackÐand returns NO.

If the object returns YES from prepareForDataSource, then the controller checks with its delegate
save and sends insertObject:, deleteObject:, or updateObject: to the data source depending on the c
any of these messages fails, this method informs the delegate, stops saving, and returns NO.

This method may invoke any of the delegate methods listed below, depending on the individual sa
performed. See the descriptions of the individual methods for more information on how they intera
saveToDataSource.

controller:object:failedToPrepareForDataSource:

controllerWillSaveToDataSource:
controllerDidSaveToDataSource:

controller:willRollbackDataSource:
controller:didRollbackDataSource:

controller:willInsertObject:inDataSource:
controller:didInsertObject:inDataSource:
controller:failedToInsertObject:inDataSource:

controller:willDeleteObject:inDataSource:
controller:didDeleteObject:inDataSource:
controller:failedToDeleteObject:inDataSource:

controller:willUpdateObject:inDataSource:
controller:didUpdateObject:inDataSource:
controller:failedToUpdateObject:inDataSource:

insertObjectAtIndex:, deleteObjectAtIndex:, endEditing

saveToDataSource:sender

This action method invokes saveToDataSource, returning self if saveToDataSource returns YES, r

(BOOL)saveToObjects

Ends editing and saves all edits made to objects as operations to apply to the controller's data sour
changes are successfully saved to the controller's objects, NO if any save operation fails or is refus
this method returns NO some changes may have been saved while others may not.

In saving a set of edits to an enterprise object, the controller first confirms the action with its deleg
controller:willSaveEdits:toObject: message if the delegate returns nil the controller aborts the save
NO. The controller takes a non-nil return value from this message and converts the new values by
forKey: to its data source. It then applies the converted values to the enterprise object by sending th

saveToObjects:sender

This action method invokes saveToObjects, returning self if saveToObjects returns YES, nil if it re

(BOOL)selectNext

Selects the object after the currently selected object. If no objects are selected or if the last object is
object becomes selected. If multiple objects are selected, this method selects the object after the fir
selection. This method results in a setSelectionIndexes: message being sent, and returns the value r
message.

This method is useful for user interfaces that display only one object at a time.

selectNext:

selectNext:sender

This action method invokes selectNext, returning self if selectNext returns YES, nil if it returns NO

(BOOL)selectPrevious

Selects the object before the currently selected object. If no objects are selected then the first objec
the first object is selected then the last object becomes selected. If multiple objects are selected, thi
object before the first object in the selection. This method results in a setSelectionIndexes: messag
returns the value returned by that message.

This method is useful for user interfaces that display only one object at a time.

selectPrevious:

selectPrevious:sender

This action method invokes selectPrevious, returning self if selectPrevious returns YES, nil if it ret

(NSArray *)selectedObjects

Returns the selected objects.

selectionIndexes

(NSArray *)selectionIndexes

Returns an array of NSNumber objects identifying the indexes of the selected objects in the contro

(void)setDataSource:(id <EODataSources>)aDataSource

Sets the controller's data source to aDataSource, clears the selection and the undo stack, and discar
This method invokes the delegate method controller:didChangeDataSource:.

releaseUndos, discardEdits, clearSelection

(void)setDelegate:anObject

Sets the controller's delegate to anObject. Doesn't retain anObject.

(void)setMarksEveryOperation:(BOOL)flag

Sets according to flag whether the controller marks an undo point for every insertion, deletion, or e
Controllers created programmatically don't mark every operation by default, while those created in

markUndo, deleteSelection

(void)setMaximumUndoMarks:(unsigned int)anInt

Sets to anInt the maximum number of undo marks that will be recorded. Marks added after the ma
causes earlier marks to be discarded. This method is useful for restricting memory usage. Setting th
allows unlimited undo queueing this is the default. Use setUndoEnabled: to disable undo.

undo

setNextController:(EOController *)aController

Sets the next controller in the controller chain to aController. See the class description for informat
chain.

(BOOL)setSavesToDataSourceAutomatically:(BOOL)flag

Sets according to flag whether operations such as insert, delete, and update with edits automaticall
performing saveToDataSource. Returns YES if successful, NO if the controller has any pending op
source.

Controllers created programmatically don't save automatically by default, but those created in Inte

setSavesToObjectsAutomatically:

(BOOL)setSavesToObjectsAutomatically:(BOOL)flag

Sets the selection as an array of NSNumber objects. Returns YES if the selection is changed to aSe

The controller recursively sends isDiscardAllowed to all of its detail controllers, and if any of them selection isn't changed. If the detail controllers allow discard this method invokes endEditing, sets and notifies its associations with selectionDidChange. Finally, this method sends controllerDidCha delegate.

If an association attempts to change the selection of its controller and fails, the association should r user interface object from the controller's selection. It can do this by simply sending itself a selecti

(void)setSelectsFirstObjectAfterFetch:(BOOL)flag

Sets according to flag whether the controller selects the first of its objects after performing a fetch.

setSelectionIndexes:

(void)setSortOrdering:(NSArray *)keySortOrderArray

Sets an EOKeySortOrder array to be applied to all records upon a fetch or resort.

sortOrdering, resort

(void)setUndoEnabled:(BOOL)flag

Enables or disables undo according to flag. Undo is by default not enabled for controllers created p though it is for controllers created in Interface Builder.

This method doesn't affect existing undo marks. You can temporarily disable undo, then reenable i however, any changes made while undo was disabled won't be reverted.

(void)setValues:(NSDictionary *)newValues forObject:anEO

Applies newValues to anEO as if they were edits made by an association. If the controller doesn't objects, the new values are buffered as edits that will actually be sent to anEO upon the next saveT the controller saves automatically to objects then the changes are immediately saved to the objects

The controller's associations are not notified when an edit is made with this method. You should s redisplay message after you finish editing objects programmatically to update the user interface.

associationDidEdit:, savesToObjectsAutomatically

setSortOrdering:, resort

NO the undo operation is aborted. Similarly, for each change reverted the controller sends its deleg
willUndoObject: message if the delegate returns NO then the undo for that object is skipped but th
continues. After the change is reverted the controller sends controller:didUndoObject: to its delega
changes have been reverted, the controller sends controllerDidUndo: to its delegate and contentsDi
associations. If the controller's selection changes it also sends selectionDidChange to its associatio

endEditing, redisplay, markUndo, setMarksEveryOperation:, setUndoEnabled:

undo:sender

This action method invokes undo and returns self.

(void)controller:(EOController *)controller
association:(EOAssociation *)anAssociation
didEditObject:anEO
key:(NSString *)aKey
value:aValue

Sent from associationDidEdit: to inform the delegate that anAssociation has performed an edit.

(void)controller:(EOController *)controllercreateObjectFailedForDataSource:dataSource

This method can be implemented to respond when the controller is unable to create an object for a
delegate does not implement this method, the controller will put up a panel to alert the user of the e
delegate is responsible for notifying the user.

(void)controller:(EOController *)controller didChangeDataSource:(id <EODataSources>)aDat

Sent from setDataSource: to inform the delegate that controller's data source is now aDataSource.

(void)controllerDidChangeSelection:(EOController *)controller

Sent whenever controller's selection changes to inform the delegate of the change.

(void)controller:(EOController *)controller didDeleteObject:anEO

Sent from any of the delete methods to inform the delegate that controller has deleted anEO.

(void)controller:(EOController *)controller

(void)controller:(EOController *)controller didInsertObject:anEO

Sent from saveToObjects to inform the delegate that controller has inserted anEO.

(void)controller:(EOController *)controller
didInsertObject:anEO
inDataSource:aDataSource

Sent from saveToDataSource to inform the delegate that controller has inserted anEO into aDataSo

(void)controller:(EOController *)controller didRollbackDataSource:(id <EODataSources>)aD

Sent from saveToDataSource to inform the delegate that controller has rolled back changes in aDa

(void)controllerDidSaveToDataSource:(EOController *)controller

Sent from saveToDataSource to inform the delegate that controller has finished saving all operatio
This method is only invoked if all operations were successfully saved.

(void)controller:(EOController *)controller didSaveToObject:anEO

Sent from saveToObjects to inform the delegate that controller has saved edits to anEO.

(void)controllerDidUndo:(EOController *)controller

Sent from undo to inform the delegate that controller has performed an undo operation.

(void)controller:(EOController *)controller didUndoObject:anEO

Sent from undo to inform the delegate that controller has undone changes to anEO.

(void)controller:(EOController *)controller
didUpdateObject:anEO
inDataSource:aDataSource

Sent from saveToDataSource to inform the delegate that controller has updated anEO in aDataSou

In rolling back the data source, the controller invokes the delegate methods controller:willRollback
controller:didRollbackDataSource:.

(EODataSourceFailureResponse)controller:(EOController *)controller
failedToInsertObject:anEO
inDataSource:aDataSource

Sent from saveToDataSource to inform the delegate that controller has failed to insert anEO into a
delegate returns EOContinueDataSourceFailureResponse controller continues saving other change
the delegate returns EORollbackDataSourceFailureResponse controller sends rollback to its data s
responds to that message. saveToDataSource then aborts and returns a failure result.

In rolling back the data source, the controller invokes the delegate methods controller:willRollback
controller:didRollbackDataSource:.

(EODataSourceFailureResponse)controller:(EOController *)controller
failedToUpdateObject:anEO
inDataSource:aDataSource

Sent from saveToDataSource to inform the delegate that controller has failed to update anEO in aD
delegate returns EOContinueDataSourceFailureResponse controller continues saving other change
the delegate returns EORollbackDataSourceFailureResponse controller sends rollback to its data s
responds to that message. saveToDataSource then aborts and returns a failure result.

In rolling back the data source, the controller invokes the delegate methods controller:willRollback
controller:didRollbackDataSource:.

(EODataSourceFailureResponse)controller:(EOController *)controller
object:anEO
failedToPrepareForDataSource:aDataSource

Sent from saveToDataSource when anEO returns NO from a prepareForDataSource message. If th
EOContinueDataSourceFailureResponse controller continues saving other changes to the data sou
returns EORollbackDataSourceFailureResponse controller sends rollback to its data source if the d
that message. saveToDataSource then aborts and returns a failure result.

(void)controller:(EOController *)controller
saveObjectsFailedForDataSource:aDataSource

Sent from saveToDataSource to inform the delegate that aDataSource returned NO from a saveObj

(void)controller:(EOController *)controller sortObjects:(NSMutableArray *)objects

If the delegate implements this method,it is responsible for sorting the given object array. If the de
implement this method and a sortOrdering has been specified, the controller will sort the objects it
sortUsingKeyOrderArray:[self sortOrdering]].

(EODataSourceOperationDelegateResponse)controller:(EOController *)controller
willDeleteObject:anEO
inDataSource:aDataSource

Sent from saveToDataSource to inform the delegate that controller will delete anEO from aDataSo
returns EODiscardDataSourceOperation the deletion isn't performed (though saveToDataSource c
operations) if the delegate returns EOPerformDataSourceOperation the deletion proceeds. A return
EOContinueDataSourceOperation indicates that the operation is not to be performed, but should be
stack to be performed on the next save. A return value of EORollbackDataSourceOperation indicat
saveToDataSource operation should be aborted and all changes rolled back.

(BOOL)controllerWillDiscardEdits:(EOController *)controller

Sent from isDiscardAllowed to inform the delegate that controller is about to discard pending edits
implement this method controller opens an attention panel warning the user that edits may be disca
returns NO then isDiscardAllowed returns NO if the delegate returns YES then isDiscardAllowed

If the delegate wants to make sure edits are preserved it can explicitly send saveToObjects to contr

(BOOL)controllerWillDiscardOperations:(EOController *)controller

Sent from isDiscardAllowed to inform the delegate that controller is about to discard pending oper
data source. If the delegate doesn't implement this method controller opens an attention panel warn
updates may be discarded. If the delegate returns NO then isDiscardAllowed returns NO if the dele
isDiscardAllowed proceeds.

If the delegate wants to make sure updates are preserved it can explicitly send saveToDataSource t

(BOOL)controllerWillFetch:(EOController *)controller

Invoked from fetch to inform the delegate that controller is about to fetch. If the delegate returns N
if the delegate returns YES the fetch proceeds.

(BOOL)controller:(EOController *)controller
willInsertObject:anEO
atIndex:(unsigned int)anIndex

Sent from insertObject:atIndex: to inform the delegate that controller will insert anEO into its array
If the delegate returns EODiscardDataSourceOperation the insertion is aborted if the delegate retur
EOPerformDataSourceOperation the insertion proceeds. A return value of EOContinueDataSource
that the operation is not to be performed, but should be left on the operation stack to be performed
return value of EORollbackDataSourceOperation indicates that the current saveToDataSource oper
aborted and all changes rolled back.

The delegate may modify anEO, usually by setting its primary key to a unique value.

saveToDataSource operation should be aborted and all changes rolled back.

(void)controller:(EOController *)controller willRollbackDataSource:(id <EODataSources>)aD

Sent from saveToDataSource to inform the delegate that controller will roll back changes in aData can't prevent this operation, but may take whatever action it needs based on this information.

(NSDictionary *)controller:(EOController *)controller
willSaveEdits:(NSDictionary *)newValues
toObject:anEO

Sent from saveToObjects to inform the delegate that controller will save edits to anEO. newValues value pairs whose keys are the names of properties belonging to anEO and whose values are the ne given for those properties. The delegate can return the values as they are, or can return a substitute to the object. If the delegate returns nil the save isn't performed and saveToObjects is aborted.

(BOOL)controllerWillSaveToDataSource:(EOController *)controller

Sent from saveToDataSource to inform the delegate that controller will saves changes to its data so returns NO the save is aborted if the delegate returns YES the save proceeds.

(BOOL)controllerWillUndo:(EOController *)controller

Sent from undo to inform the delegate that controller will undo changes. If the delegate returns NO aborted if the delegate returns YES the undo operation proceeds.

(BOOL)controller:(EOController *)controller willUndoObject:anEO

Sent from undo to inform the delegate that controller will undo changes previously made to anEO. NO the undo operation for anEO only is aborted (the larger undo operation proceeds) if the delega operation for anEO proceeds.

(EODataSourceOperationDelegateResponse)controller:(EOController *)controller
willUpdateObject:anEO
inDataSource:aDataSource

Sent from saveToDataSource to inform the delegate that controller will update anEO in aDataSour returns EODiscardDataSourceOperation the update isn't performed (though saveToDataSource co operations) if the delegate returns EOPerformDataSourceOperation the update proceeds. A return v EOContinueDataSourceOperation indicates that the operation is not to be performed, but should be stack to be performed on the next save. A return value of EORollbackDataSourceOperation indicat saveToDataSource operation should be aborted and all changes rolled back.