	initDatabase:entity:
Querying the DBModule database	entity
Accessing fetch groups and associations	getFetchGroups:
	rootFetchGroup
	fetchGroupNamed:
	addFetchGroup:
	associationForObject:
	editingAssociation
Performing transactions	fetchContentsOf:usingQualifier:
	fetchAllRecords:
	saveChanges:
	discardChanges:
	deleteRecord:
	appendNewRecord:
	insertNewRecord:
Browsing the record list	nextRecord:
	previousRecord:
Interface methods	takeValueFrom:
	textDidEnd:endChar:
	textWillChange:
	textWillEnd:
Accessing the delegate	setDelegate
	delegate:

Creates a new record and adds it to the end of the root fetch group's DBRecordList. This is a conv
implemented by sending an insertNewRecordAt: message to the root fetch group. Returns self if t
successfully appended otherwise returns nil.

insertNewRecordAt: (DBFetchGroup)

associationForObject:anObject

Returns the DBAssociation object that's associated with the given user interface object.

database

Returns the DBDatabase object for which the DBModule was created.

initDatabase:entity:

delegate

Returns the DBModule's delegate.

setDelegate:

deleteRecord:sender

Deletes the currently selected records by sending deleteCurrentSelection to the root fetch group an

deleteCurrentSelection (DBFetchGroup)

discardChanges:sender

Terminates any editing changes currently in progress for the DBModule's fetch groups. The user i
corresponding instance of DBRecordList are cleared in response to this message. All the DBAsso
notified so that they can update the display accordingly. The method is implemented by sending a
message to the DBModule's root fetch group. Returns self.

editingAssociation

Returns the DBAssociation that is currently involved in editing (the one that contains the text inser
the DBAssociation objects is involved in editing, returns nil.

entity

is closed and cannot be reopened, or if any of the fetch groups has unsaved changes that may not b

fetchContentsOf:aSource usingQualifier:aQualifier

Replaces the records in the current DBRecordList with records fetched from the database. Any ed
fetch group is terminated.

The argument aSource may be a DBEntity it may also be a DBValue that specifies a relationship.
relationship, the DBValue object contains both the key value of a source entity and the target entit
such an object responds YES to an isEntity message. For example, if the DBValue is the value ª10
ªDepartment, º the effect is to use ªDepartment = 10º as a key that defines the set of records to be f
nil, the DBModule's DBEntity is assumed.

The argument aQualifier is a DBQualifier that further restricts the records that will be fetched. If a
no further qualification and all records are returned.

If the parent DBModule's delegate responds to fetchGroupWillFetch:, it is notified. Similarly, afte
DBModule's delegate responds to fetchGroupDidFetch:, it is notified, giving it a chance to set up
DBRecordList. The various DBAssociations are notified that the contents of their views has chang
themselves. The current record index is set to 0 (the index of the first record).

Returns self when the fetch is successful, and nil otherwise. A nil return may arise if the root fetch
changes that may not be discarded.

fetchContentsOf:usingQualifier: (DBFetchGroup), isEntity (DBTypes protocol)

fetchGroupNamed:(const char *)aName

Returns the DBFetchGroup whose name matches aName (as declared in the model file or set throu
method setName:). If aName is nil, the method returns the root fetch group. Returns nil if the nam

getFetchGroups:(List *)aList

Fills aList with the DBModule's DBFetchGroup objects. Returns aList.

initDatabase:aDatabase entity:anEntity

Initializes an instance of DBModule for the given database and entity, and creates and adds the ob
Returns self.

insertNewRecord:sender

Creates a new record and inserts it into the root fetch group's DBRecordList. This is done by send
insertNewRecordAt: message to the root fetch group, passing the index of the current record as the
self if the record was successfully inserted otherwise returns nil.

insertNewRecordAt: (DBFetchGroup)

previousRecord:sender

Moves the current selection back to the previous record. However, if there is no currently selected
Returns self.

rootFetchGroup

Returns the module's one required DBFetchGroup (the first in the list of fetch groups).

saveChanges:sender

Causes all changes made within the module to be saved to the database, by saving all the module's
self, but nil if any error occurred.

Instructs the root DBFetchGroup to save the changes that the user has introduced by editing the mo
Returns self if the changes were successfully saved (or if there were no changes to save).

If the database supports transactions and no other transaction is in progress, the saveChanges: meth
new transaction before starting the save, and commits the transaction if the save is completed succ
changes within the module are saved as a single transaction (see the DBDatabase methods beginTr
commitTransaction).

If for any reason the save could not be carried out, saveChanges: returns nil, and leaves the databas
are several reasons a save might be unsuccessful. Before starting the save, the fetch groups may ru
The method also notifies the DBModule's delegate by sending it a moduleWillSave message, givin
to interpose its own checks. When the save has been carried out, the method again notifies the del
sending it a moduleDidSave message. The delegate may still object at this point if it does, the sav

setDelegate: anObject

Makes anObject the delegate of the DBModule instance. Returns self.

takeValueFrom:sender

Notifies the DBModule that the user modified one of the displays (DBImageView, NXBrowser).
the corresponding DBAssociations and through them their DBFetchGroups and causes the object's
into the appropriate part of the DBRecordList. Returns self however, if sender has no association
module's DBRecordList, returns nil.

textDidEnd:textObject endChar:(unsigned short)whyEnd

Called by a DBEditableTextFormatter object when it has relinquished first responder status. The a
identifies the character (Tab, Shift-Tab, or Return) that caused the sender to cease being first respo
permits the change to proceed a return of NO prevents the change and selects the entire text field.
not normally need to use this method explicitly.

Called by a DBEditableTextFormatter object when it is about to relinquish first responder status. A
permits the change to proceed a return of NO prevents the change and selects the entire text field.
not normally need to use this method explicitly.

moduleDidSave:module

Called when module has completed a save to the database.

(BOOL)moduleWillLoseChanges:module

Called when module is about to discard changes received from the user interface.

(BOOL)moduleWillSave:module

Called when module is about to save its data to the database.