MiscUndoManager

Inherits From:	NSObject
Declared In:	MiscUndoManager.h

Class Description

Objects of the class MiscUndoManager provide an easy way to handle undo and redo actions.   Clients of an MiscUndoManager instance make a method undoable by registering within the method an appropiate NSInvocation instance with the MiscUndoManager instance.   Registering is done by sending the message **registerRecord:** with the NSInvocation instance as argument.   Registering an NSInvocation instance adds it to the current group of records.   Undone is always a whole group of records.   The message **startUndo::** starts a new group and sending **commitUndo** adds the group to the undoArray ready to be undone.

Sending the MiscUndoManager instance the message **undo** will undo the last commited group and put the MiscUndoManager instance in a "redo" state.   Every group of records commited in this state are added to the redoArray thus providing an redo mechanism.

Since sometimes changes should be ignored (ie.   all of the discrete movements of a shape in a drag loop should not be undoable, just the final overall move), the methods **disableUndoRegistration** and **reenableUndoRegistration** can be sent to control the registration process.   For example, you might call *[undoManager disableUndoRegistration]* on a **mouseDown:** to prevent methods that register themselves with the undoManager to register each mouse drag, then you would call *[undoManager reenableUndoRegistration]* on **mouseUp:** with another call that would undo the drag loop (ie, [undoManager registerRecord:aMoveInvocation]).

There are two ways to construct the appropiate NSInvocation instances.   The two ways will be illustrated below with an example.   Suppose we have a class *Circle* with instance variables *radius* (a float) and *undoManager* (an MiscUndoManager instance) and the undoable method *-(void)setRadius:(float)aRadius*.

1.) Let the undoManager construct the NSInvocation instance.   You send the undoManager the message that will be called in the undo process and the undoManager constructs the

NSInvocation instance and registers it. Keep in mind two things: the message must be a message not implemented by MiscUndoManager instances (i.e. **respondsToSelector:** with the message selector should return NO) and the undoManager must first know the target intended for the message (set it with **setCurrentTarget:**). In our example we have:

```
                - (void)setRadius:(float)aRadius
                        {
                                ....
                                // start a new undo group
                                [undoManager startUndo:@"Undo Change Radius" :@"Redo
        Change Radius"];

                                // set the intended target for the message in the undo process
                                [undoManager setCurrentTarget:self];

                                // send the message to the undoManager. The casting is necessary
        otherwise
                                // you get a warning with "MiscUndoManager does not respond..."
                                [(id)undoManager setRadius:radius];
                                // commit group
                                [undoManager commitUndo];

                                radius = aRadius;
                                ....
                        }
```

2.) Construct the NSInvocation instance yourself. In the example method we have:

```
                - (void)setRadius:(float)aRadius
                        {
                                ...
                                NSInvocation *undoRecord;
                                SEL undoSelector;
                                NSMethodSignature *undoSignature;

                                ....
                                // start a new undo group
                                [undoManager startUndo:@"Undo Change Radius" :@"Redo
        Change Radius"];

                                // construct undoRecord
                                undoSelector = @selector(setRadius:);
                                undoSignature = [self methodSignatureForSelector:undoSelector];
```

```
                                        undoRecord = [NSInvocation
                    invocationWithMethodSignature:undoSignature];
                                        [undoRecord setSelector:undoSelector];
                                        [undoRecord setTarget:self];
                                        [undoRecord setArgument:&radius atIndex:2];
                                        [undoRecord retainArguments];

                                        // register undoRecord
                                        [undoManager registerRecord:undoRecord];

                                        // commit group
                                        [undoManager commitUndo];

                                        radius = aRadius;
                                        ....
                                }
```

The method **setLevelsOfUndo:** sets the number of levels of undo to maintain.   When the number of records exceeds this amount, the excess is removed.   Before and after each undo and redo the MiscUndoManager notifies the default notification center.

Instance Variables

No @public or @protected instance variables declared in this class.

Method Types

Starting and commiting record group	
	- startUndo::
	- commitUndo
Registering a undo record	- registerRecord:
	- setCurrentTarget:
Methods for menu validation	- undoName
	- redoName

	- numberOfUndos
	- numberOfRedos
Undo and redo	- undo
	- redo
Enabling and disabling registration	
	- disableUndoRegistration
	- reenableUndoRegistration
Setting and querying the levels of undo	
	- levelsOfUndo
	- setLevelsOfUndo:
Emptying manager	- emptyManager

Instance Methods

commitUndo

- (void)**commitUndo**

Commits the current group of MiscUndoRecords to the undoArray or if the manager is in "redo" state to the redoArray.   If no group has been started this method does nothing.

disableUndoRegistration

- (void)**disableUndoRegistration**

Disables the registration of records.   Should be used in pair with **reenableUndoRegistration**.   Can be nested.

emptyManager

- (void)**emptyManager**

Resets the manager.   The current record group is released, the undoArray and the redoArray are emptied. Appropiate for sending for example to an MiscUndoManager of a document when the document is saved.

levelsOfUndo

- (unsigned)**levelsOfUndo**

Returns the number of levels of undo the manager maintains.

numberOfRedos

- (unsigned)**numberOfRedos**

Returns the number of groups in the redoArray.   Can be used to establish whether the "Redo" menu cell should be disabled.

numberOfUndos

- (unsigned)**numberOfUndos**

Returns the number of groups in the redoArray.   Can be used to establish whether the "Undo" menu cell should be disabled.

redo

- (void)**redo**

Removes the group of MiscUndoRecords last commited to the redoArray and sends all the records in the group the message **dispatch**.   Puts the manager in the "undo" state.   Notifies the default notification center at the start of the method with the string "MiscWillRedo" and at the end of the method with "MiscDidRedo".

redoName

- (NSString *)**redoName**

Returns the redo name of the group last commited to the redoArray.   Can be used to set the title of the "Redo" menu cell.

reenableUndoRegistration

- (void)**reenableUndoRegistration**

Reenables the registration of records.   The manager is only then reenabled for registration when for every nested **disableUndoRegistration** message the manager has received a **reenableUndoRegistration** message.

registerRecord:

- (void)**registerRecord:**(NSInvocation *)*record*

Adds *record* to the current *record* group. Note the group must first be commited before the records can be undone.

setCurrentTarget:

- (void)**setCurrentTarget:***aTarget*

Sets the intended target for the message to be executed in an undo process. Does not retain *aTarget*. You must make sure that *aTarget* is still valid when you send a message to be registered (see example).

setLevelsOfUndo:

- (void)**setLevelsOfUndo:**(unsigned)*value*

Sets the number of undo levels maintained by the manager to *value*. The default number of levels is 99.

startUndo::

- (void)**startUndo:**(NSString *)*undoName* **:**(NSString *)*redoName*

Starts a new record group with undo name *undoName* and redo name *redoName*. If the current group has not been commited it is released.

undo

- (void)**undo**

Removes the group of MiscUndoRecords last commited to the undoArray and sends all the records in the group the message **dispatch**. Puts the manager in the "redo" state. Notifies the default notification center at the start of the method with the string "MiscWillUndo" and at the end of the method with "MiscDidUndo".

undoName

- (NSString *)**undoName**

Returns the undo name of the group last commited to the undoArray.   Can be used to set the title of the "Undo" menu cell.