initWith:inDocument:

Identifying objects document
editedObject
window

Displaying objects resetObject:

Managing the selection wantsSelection
selectObjects:
makeSelectionVisible:

Copying and pasting objects copySelection
deleteSelection
pasteInSelection
acceptsTypeFrom:

Opening and closing editors close
openSubeditorFor:
closeSubeditors

Activating the editor orderFront
activate

(NXAtom)acceptsTypeFrom:(const NXAtom *)typeList

Implement to return the pasteboard types your editor accepts. typeList is an array of character pointers holding the type names, with the last pointer being NULL. Each of the pointers is of the type NXAtom, meaning that the type name is a unique string. If your editor doesn't accept any of the supplied types, it should return NULL.

For example, if an editor only accepts the type IBObjectPboardType, it could implement this method in this way:

(NXAtom)acceptsTypeFrom:(const NXAtom *)typeList

orderFront

close

Implement to close the editor and free its resources. This method can be invoked for a number of r
Interface Builder invokes this method when the user closes the document. Or, your editor might se
message when the user closes the editor's window.

As part of the implementation of this method, send an editorDidClose:for: message to the active do
that this editor has closed:

editorDidClose:for: (IBDocuments protocol)

closeSubeditors

Implement to close all subeditors.

openSubeditorFor:

(BOOL)copySelection

Implement to copy the selected object(s) to the pasteboard. When the user chooses the Cut or Cop
Interface Builder, the editor that owns the selection receives a copySelection message.

In your implementation of this method, you should send the document object a copyObject: type:in
copyObjects:type:inPasteboard: message, as declared in the IBDocuments protocol. Return YES i
to the pasteboard NO otherwise.

deleteSelection

(BOOL)deleteSelection

Implement to delete the selected object(s). This method is invoked when the user deletes the selec
key or as part of the Cut command (after the selection has been copied using the copySelection me

In your implementation of this method, you should send the document object a deleteObject: or a c
as declared in the IBDocuments protocol. Return YES if the selection was deleted NO otherwise.

copySelection

Implement to return the object that's being edited. This is generally the object that the user double
editor.

initWith:anObject inDocument:aDocument

Implement this method to initialize a newly allocated editor. anObject is the object that is being ed
object that the user has double-clicked). aDocument is the currently active document, as would be
activeDocument message to NXApp. Typically, an editor object caches the document object in on
variables, since editors must frequently communicate with the document object.

makeSelectionVisible:(BOOL)flag

Implement to add or remove the selection markings from the current selection. An editor receives
makeSelectionVisible: message whenever Interface Builder wants to ensure that the selection is pr
example, when a window becomes key, the editor that owns the selection in the window receives a
YES message. When the window loses its key window status, the editor that owns the selection re
makeSelectionVisible:NO message.

openSubeditorFor:anObject

Implement to open the subeditor for anObject. An editor receives this message when the user doul
editor's selection. For the return value of this method, the editor should return nil if there is no sub
should return the id of the subeditor.

orderFront

Implement to bring the editor's window to the front. When a user double-clicks an object, the con
an orderFront and then an activate message.

activate

(BOOL)pasteInSelection

Implement to paste the object(s) from the pasteboard into the current selection. When the user cho
command in Interface Builder, the editor that owns the selection receives a pasteInSelection messa
implementation of the corresponding method should invoke the document object's pasteType:from
method.

This method should return YES if the paste operation was successful NO otherwise.

pasteType:fromPasteboard:parent: (IBDocuments protocol)

resetObject:anObject

(BOOL)wantsSelection

Implement to return YES if the editor is willing to become the selection owner NO if not.

window

Implement to return the editor window.