Enabled:(BOOL)flagSets whether the ActionCell reacts to mouse events
Bezeled:(BOOL)flagAdds or removes the ActionCell's bezel
Bordered:(BOOL)flagAdds or removes the ActionCell's border
Alignment:(int)modeSets the ActionCell's text alignment to mode
FloatingPointFormat:(BOOL)autoRangeSets the ActionCell's floating point format
left:(unsigned int)leftDigits
right:(unsigned int)rightDigits
Font:fontObjectSets the ActionCell's Font to fontObject
Icon:(const char *)iconNameSets the ActionCell's icon to the NXImage named iconName

uble)doubleValueReturns the ActionCell's contents as a double
oat)floatValueReturns the ActionCell's contents as a float
t)intValueReturns the ActionCell's contents as an int
StringValue:(const char *)aStringSets the ActionCell's contents to a copy of aString
StringValueNoCopy:(char *)aStringSets the ActionCell's contents to a aString will free the
shouldFree:(BOOL)flagstring when freed if flag is YES
nst char *)stringValueReturns the ActionCell's contents as a string

wSelf:(const NXRect *)cellFrameDraws the ActionCell in controlView
inView:controlView
ntrolViewReturns the View in which the ActionCell was most recently drawn (usually a Control)

Action:(SEL)aSelectorSets the ActionCell's action method to aSelector
EL)actionReturns the ActionCell's action method
Target:anObjectSets the ActionCell's target object to anObject
getReturns the ActionCell's target object

Tag:(int)anIntSets the ActionCell's tag to anInt
t)tagReturns the ActionCell's tag

d:(NXTypedStream *)streamReads the ActionCell from stream
te:(NXTypedStream *)streamWrites the ActionCell to stream

dNibFile:(const char *)filenameLoads objects built using Interface Builder
owner:anObject
dNibFile:(const char *)filenameLoads objects built using Interface Builder
owner:anObject
withNames:(BOOL)flag
dNibFile:(const char *)filenameLoads objects built using Interface Builder
owner:anObject
withNames:(BOOL)flag
fromZone:(NXZone *)zone
dNibSection:(const char *)nameLoads objects built using Interface Builder
owner:anObject
dNibSection:(const char *)nameLoads objects built using Interface Builder
owner:anObject
withNames:(BOOL)flag
dNibSection:(const char *)nameLoads objects built using Interface Builder
owner:anObject
withNames:(BOOL)flag
fromHeader:(const struct mach_header *)header
dNibSection:(const char *)nameLoads objects built using Interface Builder
owner:anObject
withNames:(BOOL)flag
fromZone:(NXZone *)zone
dNibSection:(const char *)nameLoads objects built using Interface Builder
owner:anObject
withNames:(BOOL)flag
fromHeader:(const struct mach_header *)header
fromZone:(NXZone *)zone
nst char *)appNameReturns the application's name
MainMenu:aMenuMakes aMenu the application's main menu
inMenuReturns the id of the application's main menu

plicationDidLaunch:appNameNotice that appName launched your arbitrary return
plicationDidTerminate:appNameNotice that appName ended your arbitrary return
plicationWillLaunch:appNameNotice that appName will launch your arbitrary return

t)activate:(int)contextNumberMakes contextNumber the active application
t)activateSelf:(BOOL)flagMakes this the active application
t)activeAppReturns context number of the active application
comeActiveAppResponds to activating the application
ctivateSelfDeactivates the application
OOL)isActiveReturns whether this is the active application
ignActiveAppResponds to deactivating the application

for:theWindowSets up a modal session with theWindow
t)runModalSession:(NXModalSession *)session
Runs a modal session
dModalSession:(NXModalSession *)sessionFinishes a modal session
ayedFree:theObjectFrees theObject after finishing the current event
OOL)isRunningReturns whether the main event loop is running
dEvent:(NXEvent *)theEventDispatches events to other objects

XEvent *)currentEventReturns pointer to the current event
XEvent *)getNextEvent:(int)maskReturns pointer to the next event matching mask
XEvent *)getNextEvent:(int)maskReturns pointer to the next event matching mask
waitFor:(double)timeout
threshold:(int)level
XEvent *)peekAndGetNextEvent:(int)maskReturns pointer to the next event matching mask
XEvent *)peekNextEvent:(int)maskReturns pointer to the next event matching mask
into:(NXEvent *)eventPtr
XEvent *)peekNextEvent:(int)maskReturns pointer to the next event matching mask
into:(NXEvent *)eventPtr
waitFor:(float)timeout
threshold:(int)level

OOL)isJournalableReturns whether the application can be journaled
Journalable:(BOOL)flagSets whether the application can be journaled
sterJournalerReturns the controlling NXJournaler object
veJournalerReturns the controlled NXJournaler object

plicationDefined:(NXEvent *)theEventResponds to an application-defined event
e:senderHides all the application's windows
OOL)isHiddenYES if windows are hidden
t)unhideResponds to message to unhide windows
hide:senderRestores hidden windows to the screen
hideWithoutActivation:senderRestores hidden windows without activating their owner
werOff:(NXEvent *)theEventResponds to a power-off subevent
t)powerOffIn:(int)ms andSave:(int)aFlagResponds to message from Workspace Manager
ntMouseDown:(NXEvent *)theEventCauses main menu to pop up under the mouse
t)unmounting:(const char *)fullPathResponds to message from Workspace Manager
ok:(int *)flag

OOL)sendAction:(SEL)aSelectorSends an action message to aTarget or up the responder
to:aTargetchain
from:sender

oSpeakerReturns the Speaker for this application
nst char *)appListenerPortNameReturns name used to register Listener with name server
rt_t)replyPortReturns port for synchronous return messages

oIconReturns the Window with the application's icon
dWindow:(int)windowNumReturns Window object corresponding to windowNum
WindowNumbers:(int **)listGets window numbers for the application's windows
count:(int *)winCount
yWindowReturns the the key window
inWindowReturns the main window
keWindowsPerform:(SEL)aSelectorSends aSelector message to the Windows
inOrder:(BOOL)flag
Autoupdate:(BOOL)flagSets whether to send Windows update messages
dateWindowsSends update message to all on-screen Windows
ndowListReturns a List of the application's Windows
niaturizeAll:Miniaturizes all the receiver's application windows
ventWindowOrderingSuppresses usual window ordering entirely

WindowsMenu:aMenuSets the Windows menu
ndowsMenuReturns the Windows menu
angeInFront:Orders all registered Windows to the front
dWindowsItem:aWindowAdds a menu item for aWindow
title:(const char *)aString
filename:(BOOL)isFilename
moveWindowsItem:aWindowRemoves the Window's menu item
ngeWindowsItem:aWindowChanges the Window's menu item
title:(const char *)aString
filename:(BOOL)isFilename
dateWindowsItem:aWindowUpdates the Window's menu item

wHelpPanel:senderShows the application's help panel or default
erFrontDataLinkPanel:Shows the shared instance creates if need be

ServicesMenu:aMenuSets the Services menu
vicesMenuReturns the Services menu
isterServicesMenuSendTypes:(const char *const *)sendTypes
andReturnTypes:(const char *const *)returnTypes
Registers pasteboard types the application can send/receive
idRequestorForSendType:Indicates whether the Application can send and receive the
(NXAtom)sendTypespecified types
andReturnType:(NXAtom)returnType

PSContext)contextReturns the Application's DPS context
usViewGets the currently lockFocus'ed View
nst char *)hostNameReturns machine running the Window Server

nst char *const *)systemLanguagesGets a list of the user's preferred languages

t)openFile:(const char *)fullPathAsks the delegate to open the fullPath file
ok:(int *)flag
t)openTempFile:(const char *)fullPathAsks the delegate to open a temporary file
ok:(int *)flag
t)fileOperationCompleted:(int)operationNotification of completion of NXWorkspaceRequest arbitrary operati

)mounted:(const char *)fullPathNotice that device at fullPath has been mounted returns 0 (unless overrideen i
mounted:(const char *)fullPathNotice that device at fullPath has been unmounted returns 0 (unless overrideen

PrintInfo:infoMakes info the application's PrintInfo object
ntInfoReturns the application's PrintInfo object
PageLayout:senderRuns the application's PageLayout panel

erFrontColorPanel:senderBrings up the color panel
esImportAlphaYES if application responds to opacity in imported colors
ImportAlpha:Enable/disable response to opacity in imported colors

minate:senderFrees the Application object and exits the application

Delegate:anObjectMakes anObject the Application's delegate
egateReturns the Application's delegate

:sender Notice that appName will launch²
applicationWillLaunch:(const char *)appName
:sender Notice that appName launched²
applicationDidLaunch:(const char *)appName
WillInit:senderNotifies delegate before initializing²

t)app:senderOpens filename²
openFile:(const char *)filename
type:(const char *)aType
t)app:senderOpens temporary file filename²
openTempFile:(const char *)filename
type:(const char *)aType
XDataLinkManager *)app:sender Open application to run without user interface²
openFileWithoutUI:(const char *)filename
type:(const char *)aType
p:sender Notice that operation completed your arbitrary return²
fileOperationCompleted:(int)operation
p:sender mounted:(const char *)fullPathNotification that device at fullPath was mounted²
t)app:senderFacilitates unmounting a device²
unmounting:(const char *)fullPath
p:sender Notification that device at fullPath was unmounted²
unmounted:(const char *)fullPath
pWillTerminate:senderNotification that application will terminate²
p:senderNotification that sender will show panel
willShowHelpPanel:panel
p:senderResponds to message from Workspace Manager²
powerOffIn:(int)ms
andSave:(int)aFlag
werOff:(NXEvent *)theEventResponds to a poweroff subevent
p:sender Notification that application terminated²
applicationDidTerminate:(const char *)appName
ese methods may be defined in the delegate and/or in a subclass. If a method is implemented both in a subcla
the message is sent to the delegate.

tFrame:(const NXRect *)frameRectInitializes a new Box object with the given frameRect
eDeallocates the Box

BorderType:(int)aTypeSets the Box's border to aType
t)borderTypeReturns the Box's border type
TitlePosition:(int)aPositionSets the position of the title to aPosition
t)titlePositionReturns the position of the title
Title:(const char *)aStringSets the Box's title to aString
nst char *)titleReturns the title of the Box
lReturns the Cell used to draw the title
Font:fontObjSets the Font of the title to fontObj
tReturns the Font used to draw the title

laceSubview:aView with:anotherViewReplaces aView with anotherView within the content view

FrameFromContentFrame:(const NXRect *)contentFrame
Resizes the Box to accommodate contentFrame
eTo:(NXCoord)width :(NXCoord)heightResizes the Box to width and height
eToFitResizes the Box to exactly enclose its subviews

wSelf:(const NXRect *)rects :(int)rectCountDraws the Box

akeLays out the graphic elements of the Box
d:(NXTypedStream *)streamReads the Box object from the typed stream
te:(NXTypedStream *)streamWrites the Box object to the typed stream

Initializes a new Button with title ªButtonº
Frame:(const NXRect *)frameRectInitializes a new Button within frameRect
Frame:(const NXRect *)frameRectInitializes a new Button within frameRect,
icon:(const char *)iconNamewith the NXImage named iconName as its icon,
tag:(int)anIntanInt as its tag,
target:anObjectanObject as its target object,
action:(SEL)aSelectoraSelector as its action message,
key:(unsigned short)charCodea key equivalent of charCode,
enabled:(BOOL)flagand enabled according to flag
Frame:(const NXRect *)frameRectInitializes a new Button within frameRect,
title:(const char *)aStringwith aString as its title,
tag:(int)anIntanInt as its tag,
target:anObjectanObject as its target object,
action:(SEL)aSelectoraSelector as its action message,
key:(unsigned short)charCodea key equivalent of charCode,
enabled:(BOOL)flagand enabled according to flag

Type:(int)aTypeSets how the Button highlights and shows its state

State:(int)valueSets the Button's state to value (0 or 1)

Title:(const char *)aStringMakes aString the Button's title
TitleNoCopy:(const char *)aStringMakes aString the Button's title without copying it
nst char *)titleReturns the Button's title
AltTitle:(const char *)aStringMakes aString the Button's alternate title
nst char *)altTitleReturns the Button's alternate title

Icon:(const char *)iconNameMakes the NXImage named iconName the Button's icon
Icon:(const char *)iconNameSets the icon by name, and its position
position:(int)aPosition
nst char *)iconReturns the name of the Button's icon
AltIcon:(const char *)iconNameMakes the NXImage named iconName the alternate icon
nst char *)altIconReturns the name of the Button's alternate icon
Image:imageMakes the NXImage image the Button's icon
ageReturns the Button's image
AltImage:altImageMakes the NXImage image the alternate icon
ImageReturns the Button's alternate image
IconPosition:(int)aPositionSets the position of the Button's icon
t)iconPositionReturns the position of the Button's icon

Transparent:(BOOL)flagSets whether the Button is transparent
OOL)isTransparentReturns whether the Button is transparent
Bordered:(BOOL)flagSets whether the Button has a bezeled border
OOL)isBorderedReturns whether the Button has a bezeled border

playDisplays the Button
hlight:(BOOL)flagHighlights (or unhighlights) the Button according to flag

KeyEquivalent:(unsigned short)charCodeMakes charCode the Button's key equivalent
signed short)keyEquivalentReturns the Button's key equivalent

OOL)acceptsFirstMouseEnsures that the Button accepts first mouse-down
formClick:senderSimulates the user clicking the Button
OOL)performKeyEquivalent:(NXEvent *)theEvent
Simulates a mouse click, if the key is right

Sound:soundObjectSets the Sound played when the Button is pressed

IconCell:(const char *)iconNameInitializes a new ButtonCell with an NXImage named iconName as its icon
yFromZone:(NXZone *)zoneReturns a copy of the ButtonCell allocated from zone
eDeallocates the ButtonCell

cCellSize:(NXSize *)theSizeCalculates and returns the size of the ButtonCell
inRect:(const NXRect *)aRect
DrawRect:(NXRect *)theRectReturns the rectangle the ButtonCell draws in
TitleRect:(NXRect *)theRectReturns the rectangle the title is drawn in
IconRect:(NXRect *)theRectReturns the rectangle the icon is drawn in

Title:(const char *)aStringMakes a copy of aString the ButtonCell's title
TitleNoCopy:(const char *)aStringMakes aString the ButtonCell's title without copying it
nst char *)titleReturns the ButtonCell's title
AltTitle:(const char *)aStringMakes a copy of aString the ButtonCell's alternate title
nst char *)altTitleReturns the ButtonCell's alternate title
Font:fontObjectSets the Font used to draw the title

Icon:(const char *)iconNameMakes the NXImage named iconName the ButtonCell's icon
nst char *)iconReturns the name of the ButtonCell's icon
AltIcon:(const char *)iconNameMakes the NXImage named iconName the ButtonCell's alternate icon
nst char *)altIconReturns the name of the ButtonCell's alternate icon
Image:imageMakes the NXImage object image the ButtonCell's icon
ageReturns the ButtonCell's icon
AltImage:altImageMakes the NXImage object image the alternate icon
mageReturns the ButtonCell's alternate icon
IconPosition:(int)aPositionSets the position of the ButtonCell's icon to aPosition
t)iconPositionReturns the position of the ButtonCell's icon

Sound:aSoundSets the Sound played by the ButtonCell on a mouse-down
undReturns the Sound played by the ButtonCell

DoubleValue:(double)aDoubleSets the ButtonCell's state (value) to aDouble
uble)doubleValueReturns the ButtonCell's state as a double
FloatValue:(float)aFloatSets the ButtonCell's state (value) to aFloat
oat)floatValueReturns the ButtonCell's state as a float
IntValue:(int)anIntSets the ButtonCell's state (value) to anInt
t)intValueReturns the ButtonCell's state as an int
StringValue:(const char *)aStringSets the ButtonCell's state (value) to a copy of aString

OOL)trackMouse:(NXEvent *)theEventPlays the Sound, then tracks the mouse
inRect:(const NXRect *)cellFrame
ofView:controlView

KeyEquivalent:(unsigned short)charCodeSets the ButtonCell's key equivalent
KeyEquivalentFont:fontObjSets the Font used to draw the key equivalent
KeyEquivalentFont:(const char *)fontNameSets the Font and size used to draw the key equivalent
size:(float)fontSize
signed short)keyEquivalentReturns the ButtonCell's key equivalent

Parameter:(int)aParameter to:(int)valueSets various flag values
t)getParameter:(int)aParameterReturns various flag values

Bordered:(BOOL)flagSets whether the ButtonCell has a bezeled border
OOL)isBorderedReturns whether the ButtonCell has a bezeled border
Transparent:(BOOL)flagSets whether the ButtonCell is transparent
OOL)isTransparentReturns whether the ButtonCell is transparent
OOL)isOpaqueReturns whether receiver is opaque

Type:(int)aTypeSets the ButtonCell's display behavior
HighlightsBy:(int)aTypeSets how the ButtonCell highlights
t)highlightsByReturns how the ButtonCell highlights
ShowsStateBy:(int)aTypeSets how the ButtonCell shows its alternate state
t)showsStateByReturns how ButtonCell shows its alternate state

formClick:senderSimulates clicking the ButtonCell

wInside:(const NXRect *)aRectDraws the inside of the ButtonCell
inView:controlView
wSelf:(const NXRect *)cellFrameDraws the ButtonCell
inView:controlView
hlight:(const NXRect *)cellFrameHighlights the ButtonCell
inView:controlView
lit:(BOOL)flag

tInitializes a new Cell
tIconCell:(const char *)iconNameInitializes a new Cell with the NXImage named iconName
tTextCell:(const char *)aStringInitializes a new Cell with title aString
pyFromZone:(NXZone *)zoneReturns a copy of the receiving Cell from zone
eDeallocates the Cell

cCellSize:(NXSize *)theSizeReturns the minimum size needed to display the Cell
cCellSize:(NXSize *)theSizeReturns the minimum size needed to display the Cell
inRect:(const NXRect *)aRect
cDrawInfo:(const NXRect *)aRectImplemented by subclasses to recalculate drawing sizes
DrawRect:(NXRect *)theRectReturns the rectangle the Cell draws in
IconRect:(NXRect *)theRectReturns the rectangle that an icon is drawn in
TitleRect:(NXRect *)theRectReturns the rectangle that a title is drawn in

Type:(int)aTypeSets the Cell's type to aType
t)typeReturns the Cell's type

State:(int)valueSets the state of the Cell to value (0 or 1)
rementStateIncrements the state of the Cell
t)stateReturns the state of the Cell (0 or 1)

Enabled:(BOOL)flagSets whether the Cell reacts to mouse events
OOL)isEnabledReturns whether the Cell reacts to mouse events

Icon:(const char *)iconNameSets the Cell's icon to the NXImage named iconName
nst char *)iconReturns the name of the Cell's icon

DoubleValue:(double)aDoubleSets the Cell's value to aDouble
uble)doubleValueReturns the Cell's value as a double
FloatValue:(float)aFloatSets the Cell's value to aFloat
oat)floatValueReturns the Cell's value as a float
IntValue:(int)anIntSets the Cell's value to anInt
t)intValueReturns the Cell's value as an int
StringValue:(const char *)aStringSets the Cell's value to a copy of aString
StringValueNoCopy:(const char *)aStringSets the Cell's value to aString
StringValueNoCopy:(char *)aStringSets the Cell's value to aString will free the string when
shouldFree:(BOOL)flagfreed if flag is YES

Alignment:(int)modeSets the alignment of text in the Cell to mode
t)alignmentReturns the alignment of text in the Cell
Font:fontObjectSets the Font used to display text in the Cell to fontObject
tReturns the Font used to display text in the Cell
Editable:(BOOL)flagSets whether the Cell's text is editable
OOL)isEditableReturns whether the Cell's text is editable
Selectable:(BOOL)flagSets whether the Cell's text is selectable
OOL)isSelectableReturns whether the Cell's text is selectable
Scrollable:(BOOL)flagSets whether the Cell scrolls to follow typing
OOL)isScrollableReturns whether the Cell scrolls to follow typing
TextAttributes:textObjectSets Text parameters for drawing or editing
Wrap:(BOOL)flagSets whether the Cell's text is word-wrapped

t:(const NXRect *)aRectAllows text editing in response to a mouse-down event
inView:aView
editor:textObject
delegate:anObject
event:(NXEvent *)theEvent
dEditing:textObjectEnds any text editing occurring in the Cell
ect:(const NXRect *)aRectAllows text selection in response to a mouse-down event
inView:aView
editor:textObject
delegate:anObject
start:(int)selStart
length:(int)selLength

EntryType:(int)aTypeSets the type of data the user can type into the Cell
t)entryTypeReturns the type of data the user can type into the Cell
OOL)isEntryAcceptable:(const char *)aStringReturns whether aString is acceptable for the entry type

FloatingPointFormat:(BOOL)autoRangeSets the display format for floating point values
left:(unsigned)leftDigits
right:(unsigned)rightDigits

Bezeled:(BOOL)flagSets whether the Cell has a bezeled border
OOL)isBezeledReturns whether the Cell has a bezeled border
Bordered:(BOOL)flagSets whether the Cell has a plain border
OOL)isBorderedReturns whether Cell has a plain border
OOL)isOpaqueReturns whether the Cell is opaque

wSelf:(const NXRect *)cellFrameDraws the Cell in aView
inView:aView
hlight:(const NXRect *)cellFrameHighlights the Cell according to flag in aView
inView:aView
lit:(BOOL)flag
OOL)isHighlightedReturns whether the Cell is highlighted

Action:(SEL)aSelectorImplemented by subclasses to set the action method
EL)actionImplemented by subclasses to return the action method
Target:anObjectImplemented by subclasses to set the target object
getImplemented by subclasses to return the target object
Continuous:(BOOL)flagSets whether the Cell continuously sends action
OOL)isContinuousReturns whether the Cell continuously sends action
t)sendActionOn:(int)maskDetermines when the action is sent while tracking

Tag:(int)anIntImplemented by subclasses to set an identifier tag
t)tagImplemented by subclasses to return the identifier tag

signed short)keyEquivalentImplemented by subclasses to return a key equivalent

t)mouseDownFlagsReturns the event flags set at the start of mouse tracking
PeriodicDelay:(float*)delayReturns repeat values for continuous sending of the action
andInterval:(float*)interval
OOL)trackMouse:(NXEvent *)theEventControls tracking behavior of the Cell
inRect:(const NXRect *)cellFrame
ofView:aView
OOL)startTrackingAt:(const NXPoint *)startPoint
inView:aViewDetermines whether tracking should begin based on startPoint within aView
OOL)continueTracking:(const NXPoint *)lastPoint
at:(const NXPoint *)currentPointReturns whether tracking should continue based on
inView:aViewlastPoint and currentPoint within aView
pTracking:(const NXPoint *)lastPointAllows the Cell to update itself to end tracking, based on
at:(const NXPoint *)stopPointlastPoint, stopPoint, within aView flag is YES if the
inView:aViewthis method was invoked because mouse went up
mouseIsUp:(BOOL)flag

etCursorRect:(const NXRect *)cellFrameSets text Cells to show I-beam cursor
inView:aView

tFrame:(const NXRect *)frameRectInitializes a new ClipView instance
eReleases the ClipView's storage

veTo:(NXCoord)x :(NXCoord)yMoves the origin of the frame rectangle
ateTo:(NXCoord)angleOverridden to disable rotation
eTo:(NXCoord)width :(NXCoord)heightResizes the ClipView's frame

ate:(NXCoord)angleOverriden to disable rotation
le:(NXCoord)x :(NXCoord)yRescales the coordinate system
DrawOrigin:(NXCoord)x :(NXCoord)ySets the origin of the coordinate system
DrawRotation:(NXCoord)angleDisables rotation of the coordinate system
DrawSize:(NXCoord)width :(NXCoord)heightScales the coordinate system
nslate:(NXCoord)x :(NXCoord)yShifts the coordinate system

cViewReturns the ClipView's document view
DocView:aViewMakes aView the ClipView's document view
DocRect:(NXRect *)aRectReturns the document rectangle
DocVisibleRect:(NXRect *)aRectGets the visible portion of the document view
etCursorRectsResets the cursor rectangle for the document view
DocCursor:anObjSets the cursor for the document view

oat)backgroundGrayReturns the ClipView's background gray
BackgroundGray:(float)valueSets the ClipView's background gray
XColor)backgroundColorReturns the ClipView's background color
BackgroundColor:(NXColor)colorSets the ClipView's background color
wSelf:(const NXRect *)rects :(int)rectCountFills the background gray where needed

oscroll:(NXEvent *)theEventScrolls in response to mouse-dragged events
nstrainScroll:(NXPoint *)newOriginPrevents scrolling to an undesirable position
wScroll:(const NXPoint *)newOriginLowest-level unconstrained scrolling routine
CopyOnScroll:(BOOL)flagSets how the visible areas are redrawn
DisplayOnScroll:(BOOL)flagSets how the document view is displayed during scrolling

lectScroll:aClipViewNotifies the superview to update indicators
ollClip:aClipView to:(const NXPoint *)aPoint
Notifies the superview of a scroll

tFrame:(const NXRect *)frameRectInitializes a new Control
eDeallocates the Control

Cell:aCellSets the Control's Cell to aCell
lReturns the Control's Cell

Enabled:(BOOL)flagSets whether the Control reacts to mouse events
OOL)isEnabledReturns whether the Control reacts to mouse events

ectedCellReturns the Control's selected Cell
t)selectedTagReturns the tag of the Control's selected Cell

DoubleValue:(double)aDoubleSets the Control's value to aDouble
uble)doubleValueReturns the Control's value as a double
FloatValue:(float)aFloatSets the Control's value to aFloate
oat)floatValueReturns the Control's value as a float
IntValue:(int)anIntSets the Control's value to anInt
t)intValueReturns the Control's value as an int
StringValue:(const char *)aStringSets the Control's value to aString
StringValueNoCopy:(const char *)aStringSets the Control's value to aString
StringValueNoCopy:(char *)aStringSets the Control's value to aString
shouldFree:(BOOL)flag
nst char *)stringValueReturns the Control's value as a string

eDoubleValueFrom:senderSets the Control's value to sender's doubleValue
eFloatValueFrom:senderSets the Control's value to sender's floatValue

FloatingPointFormat:(BOOL)autoRangeSets the display format for floating point values
left:(unsigned)leftDigits
right:(unsigned)rightDigits

ortEditingAborts editing of text displayed by the Control
rentEditorReturns the object used to edit text in the Control
idateEditingValidates the user's changes to editable text

etCursorRectsSets text-bearing Controls to show an I-beam cursor

cSizeRecalculates internal size information
eTo:(NXCoord)width :(NXCoord)heightResizes the Control to width and height
eToFitResizes the Control to fit its Cell

wCell:aCellRedraws aCell if it's the Control's Cell
wCellInside:aCellRedraws aCell's inside if it's the Control's Cell
wSelf:(const NXRect *)rects :(int)rectCountDraws the Control
ectCell:aCellSelects aCell if it's the Control's cell
dateRedisplays the Control or marks it for later redisplay
dateCell:aCellRedisplays aCell or marks it for redisplay
dateCellInside:aCellRedisplays the inside of aCell or marks it for redisplay

Action:(SEL)aSelectorSets the Control's action method to aSelector
EL)actionReturns the Control's action method
Target:anObjectSets the Control's target object to anObject
getReturns the Control's target object
Continuous:(BOOL)flagSets whether the Control continuously sends its action
OOL)isContinuousReturns whether the Control continuously sends its action
dAction:(SEL)theAction to:theTargetHas the Application object send theAction to theTarget
t)sendActionOn:(int)maskDetermines when the action is sent while tracking

Tag:(int)anIntSets the Control's tag to anInt
t)tagReturns the Control's tag

oreMultiClick:(BOOL)flagSets whether multiple clicks are ignored

eHas no effect

nst float *)displayNameReturns the full name of the font
nst float *)familyNameReturns the name of the font's family
nst char *)nameReturns the name of the font
t)fontNumReturns the Window Server's font number
oat)getWidthOf:(const char *)stringReturns the width of string in this font
OOL)hasMatrixReturns whether font differs from identity matrix
nst float *)matrixReturns a pointer to the font matrix
XFontMetrics *)metricsReturns pointer to a record of font information
oat)pointSizeReturns the size of the font in points
XFontMetrics *)readMetrics:(int)flagsReads flags information into the font record
eenFontReturns the screen font for this font
t)styleReturns the font style

Makes this font the graphic state's current font
Style:(int)aStyleSets the Font's style
etUserFixedPitchFont:(Font *)aFontSets the fixed-pitch font used by default in the application
etUserFont:(Font *)aFontSets the standard font used by default in the application

nvertFont:fontObjConverts the font in response to changeFont:
nvertWeight:(BOOL)upFlag of:fontObjRaises or lowers the weight of the font
nvert:fontObj toFace:(const char *)typefaceConverts the font to the specified typeface
nvert:fontObj toFamily:(const char *)familyConverts the font to the specified family
nvert:fontObj toSize:(float)sizeConverts the font to the specified point size
nvert:fontObjConverts the font to have the specified trait
toHaveTrait:(NXFontTraitMask)trait
nvert:fontObjConverts the font to remove the specified trait
toNotHaveTrait:(NXFontTraitMask)trait
dFont:(const char *)familyTries to find a font that matches the specified characteristics
traits:(NXFontTraitMask)traits
weight:(int)weight
size:(float)size
Family:(const char **)familyProvides the characteristics of the given fontObj
traits:(NXFontTraitMask *)traits
weight:(int *)weight
size:(float *)size
ofFont:fontObj

Action:(SEL)aSelectorSets the action sent by the FontManager
etFontPanelFactory:classIdSets the class used to create the Font panel
etFontManagerFactory:classIdSets the class used to create the font manager
SelFont:fontObj isMultiple:(BOOL)flagNotifies FontManager of selection's current font
Enabled:(BOOL)flagEnables and disables the Font panel and menu

EL)actionGets the action sent by the FontManager
ar **)availableFontsProvides a list of all available fonts
FontMenu:(BOOL)createReturns the Font menu
FontPanel:(BOOL)createReturns the Font panel
OOL)isMultipleReturns whether selection contains multiple fonts
FontReturns the first font in the current selection
OOL)isEnabledReturns whether the Font panel and menu are enabled

difyFont:senderConverts current selection's font
lFontTrait:senderCauses trait to be added to font in current selection
noveFontTrait:senderCauses trait to be removed from font in current selection