

convertFont:

convertWeight:of:
convert:toFace:
convert:toFamily:
convert:toSize:
convert:toHaveTrait:
convert:toNotHaveTrait:
findFont:traits:weight:size:
getFamily:traits:weight:size:ofFont:

Setting parameters setAction:

+ setFontPanelFactory:
+ setFontManagerFactory:
setSelFont:isMultiple:
setEnabled:

Querying parameters action

availableFonts

createFont and archi
sendAction

Assigning a delegate setDelegate:

delegate

Archiving the FontManager finishUnarchiving

setFontPanelFactory:

setFontManagerFactory:

(SEL)action

Returns the action that's sent to the first responder when the user selects a new font from the Font menu.

setAction:

`NX_ADDTRAIT:` When the `convertFont:` message is received, the `FontManager` converts the supplied font to a `convert:toHaveTrait:` message.

`removeFontTrait:`, `convertFont:`, `convert:toHaveTrait:`, `selectedTag` (`Control`)

`(char **)availableFonts`

Returns by reference a NULL-terminated list of NULL-terminated PostScript font names of all the fonts available on the system as reported by the Window Server. The returned names are suitable for creating new Fonts using the `newFont:` message or the `Font` class. The fonts are not in any guaranteed order, but no font name is repeated in the list. It is the caller's responsibility to free the list when finished with it.

`convert:fontObj toFace:(const char *)typeface`

Returns a Font object whose traits are the same as those of `fontObj` except as specified by `typeface`. If `typeface` can't be made, the method returns `fontObj` itself. This method can be used to convert a font, or it can be used to convert fonts in a different manner.

`convert:toFamily:`, `convert:toSize:`, `convert:toHaveTrait:`, `convert:toNotHaveTrait:`, `convertWeight:`

`convert:fontObj toFamily:(const char *)family`

Returns a Font object whose traits are the same as those of `fontObj` except as specified by `family`. If `family` can't be made, the method returns `fontObj` itself. This method can be used to convert a font, or it can be used to convert fonts in a different manner.

`convert:toFace:`, `convert:toSize:`, `convert:toHaveTrait:`, `convert:toNotHaveTrait:`, `convertWeight:`

`convert:fontObj toHaveTrait:(NXFontTraitMask)traits`

Returns a Font object whose traits are the same as those of `fontObj` except as altered by the addition of `traits`. Of course, conflicting traits (such as `NX_CONDENSED` and `NX_EXPANDED`) have their counterparts turned other off. If the conversion can't be made, the method returns `fontObj` itself. This method can be used to convert fonts in a different manner.

`convert:toNotHaveTrait:`, `convert:toFace:`, `convert:toSize:`, `convert:toFamily:`, `convertWeight:`

`convert:fontObj toNotHaveTrait:(NXFontTraitMask)traits`

Returns a Font object whose traits are the same as those of `fontObj` except as altered by the removal of `traits`. If the conversion can't be made, the method returns `fontObj` itself. This method can be used to convert fonts in a different manner.

`convert:toHaveTrait:`, `convert:toFace:`, `convert:toSize:`, `convert:toFamily:`, `convertWeight:`

convertFont:fontObj

Converts fontObj according to the user's selections from the Font panel or menu. Whenever an ob changeFont: message from the FontManager, it should send a convertFont: message for each font i

This method determines what to do to the fontObj by checking the whatToDo instance variable and appropriate conversion method. Returns the converted font.

convertWeight:(BOOL)upFlag of:fontObj

Attempts to increase (if upFlag is YES) or decrease (if upFlag is NO) the weight of the font specified can, it returns a new font object with the higher (or lower) weight. If it can't, it returns fontObj itself. This method converts the weight only if it can maintain all of the traits of the original fontObj. This method is used to convert fonts in a different manner.

convert:toHaveTrait:, convert:toNotHaveTrait:, convert:toFamily:

delegate

Returns the FontManager's delegate.

setDelegate:

findFont:(const char *)family
traits:(NXFontTraitMask)traits
weight:(int)weight
size:(float)size

If there's a font on the system with the specified family, traits, weight, and size, then it's returned. If NX_BOLD or NX_UNBOLD is one of the traits, weight is ignored.

finishUnarchiving

Finishes the unarchiving task by instantiating the one application-wide instance of the FontManager.

getFamily:(const char **)family
traits:(NXFontTraitMask *)traits
weight:(int *)weight
size:(float *)size
ofFont:fontObj

For the given font object fontObj, copies the font family, traits, weight, and point size information referred to by this method's arguments.

getFontPanel:(BOOL)create

Returns the FontPanel that will be used when the user chooses the Font Panel command from the Font menu. If the flag is YES, the FontPanel is created if it doesn't already exist.

Unless you've specified a different class (by sending a setFontPanelFactory: message to the FontManager object), an object of the FontPanel class is returned.

getFontMenu:

(BOOL)isEnabled

Reports whether the controls in the Font panel and the commands in the Font menu are enabled or disabled.
setEnabled:

(BOOL)isMultiple

Returns whether the currently selected text has multiple fonts.

setFont:isMultiple:

modifyFont:sender

Causes the FontManager's action message (by default, changeFont:) to be sent up the responder chain. When the responder replies with a convertFont: message, the font is converted in a way specified by the selected command of this message. The Larger, Smaller, Heavier, and Lighter commands in the Font menu invoke the

addFontTrait:, removeFontTrait:

modifyFontViaPanel:sender

Causes the FontManager's action message (by default, changeFont:) to be sent up the responder chain. When the responder replies with a convertFont: message, the FontManager sends a panelConvertFont: message to the FontPanel for the conversion.

This message is almost always sent by a Control in the Font panel itself. Usually, the panel uses the FontPanel's convert routines to do the conversion based on the choices the user has made.

panelConvertFont: (FontPanel)

orderFrontFontPanel:sender

Sends orderFront: to the FontPanel. If there's no Font panel yet, a new message is sent to the FontManager to create the object you specified with the FontManager's setFontPanelFactory: class method.

convertFont:, convert:toHaveTrait:, selectedTag (Control)

setFont

Returns the last font set with setSetFont:isMultiple:.

If you receive a changeFont: message from the FontManager and want to find out what font the user selected, use the following (assuming theFontManager is the application's FontManager object).

```
setFont:isMultiple:, modifyFont:
```

sendAction

Sends the FontManager's action message (by default, changeFont:) up the responder chain. The message is sent to the FontManager object regardless of which user-interface object initiated the sending of the action. The traitToChange variables should be set appropriately before sending a sendAction message.

You rarely, if ever, need to send a sendAction message or to override this method. The message is sent to the FontManager object regardless of which user-interface objects that allow users to manipulate the font of the text (for example, the Font panel and the Font menu).

```
setAction:
```

setAction:(SEL)aSelector

Sets the action that's sent when the user selects a new font from the Font panel or from the Font menu. The action is changeFont:.

```
sendAction
```

setDelegate:anObject

Sets the FontManager's delegate. The delegate can restrict which font names appear in the Font panel. The delegate is nil by default.

setEnabled:(BOOL)flag

Sets whether the controls in the Font panel and the commands in the Font menu are enabled or disabled. When the flag is YES, the controls and commands are enabled. Even when disabled, the Font panel allows the user to preview the selected font. When the Font panel is disabled, the user can't apply the selected font to text in the application's main document.

You can use this method to disable the user interface to the font selection system when its actions are not applicable. For example, you might disable the font selection system when your application has no document open.

```
isEnabled
```

selfFont, isMultiple

(BOOL)fontManager:sender willIncludeFont:(const char *)fontName

Responds to a message informing the FontManager's delegate that the FontPanel is about to include a font that will be displayed to the user. fontName is the name of the font, for example "Helvetica- Narrow-Bold". If YES, the font is added; otherwise, it is not.

A delegate that implements this method can receive multiple fontManager:willIncludeFont: messages. The FontPanel needs updating, such as when the user selects a different family name to determine which typefaces are available. For each typeface within that family, the delegate will receive notification. Consequently, your implementation shouldn't take long to execute.