NSSelection

Inherits From:	NSObject
Conforms To:	NSCoding NSCopying NSObject (from NSObject)
Declared In:	AppKit/NSSelection.h

Class Description

The NSSelection class defines an object that describes a selection within a document. An NSSelection, or simply, selection, is an immutable description; it may be held by the system or other documents, and it cannot change over time. Selections are typically used by NSDataLink objects to represent the source and destination of a link.

Because a selection description can't be changed once it's been exported, it's a good idea to construct general descriptions that can survive changes to a document and don't require selection-specific information to be stored in the document. This description may be simple or complex, depending upon the application. For example, a painting application might describe a selection in an image as a simple rectangle. This description doesn't require that any information be stored in the image's file, and the description can be expected to remain valid through the life of the image. An object-based drawing application might describe a selection as a list of object identifiers (though *not* **id**s), where an object identifier is unique throughout the life of the document. Based on this list, a selection could be meaningfully reconstructed, even if new objects are added to the document or selected objects are deleted. Such a scheme doesn't require that any selection-specific information be stored in the document's file, with the benefit that links can be made to read-only documents.

Maintaining a character-range selection in a text document is more problematic. A possible solution is to insert selection-begin and selection-end markers that define a specific selection into the text stream. A selection description would then refer to a specific selection marker. This solution requires that selection state information be stored and maintained within the document. Furthermore, this information generally shouldn't be purged from the document, because the document can't know how many references to the selection exist. (References to the selection could be stored with documents on removable media, like floppy disks.) This selection-state information should be maintained as long as it refers to any meaningful data. For this reason, it's desirable to describe selection in a manner that doesn't require that selection-state information be maintained in the document whenever possible.

Three well-known selection descriptions can apply to any document: the empty selection, the entire document, and the abstract concept of the current selection. NSSelection objects for these selections are returned by the **emptySelection**, **allSelection**, and **currentSelection** class methods.

Since an NSSelection may be used in a document that is read by machines with different architectures, care should be taken to write machine-independent descriptions. For example, using a binary structure as a selection description will fail on a machine where an identically-defined structure has a different size or is kept in memory with different byte ordering. Exporting (and then parsing) ASCII descriptions is often a good solution. If binary descriptions must be used, it’s prudent to preface the description with a token specifying the description’s byte ordering.

It may also be prudent to version-stamp selection descriptions, so that old selections can be accurately read by updated versions of an application.

Adopted Protocols

NSCoding	– encodeWithCoder: – initWithCoder:
NSCopying	– copyWithZone:

Method Types

Creating an NSSelection	+ selectionWithDescriptionData: – initWithDescriptionData: – initWithPasteboard:
Returning special selection shared instances	+ allSelection + currentSelection + emptySelection
Describing a selection	– descriptionData – isWellKnownSelection
Writing a selection to the pasteboard	– writeToPasteboard:

Class Methods

allSelection

+ (NSSelection *)**allSelection**

Returns the shared instance of the well-known selection representing an entire document.

currentSelection

+ (NSSelection *)**currentSelection**

Returns the shared instance of the well-known selection representing the abstract concept of the current selection. The current selection never describes a specific selection; it describes a selection that may change frequently.

emptySelection

+ (NSSelection *)**emptySelection**

Returns the shared instance of the well-known selection representing no data.

selectionWithDescriptionData:

+ (NSSelection *)**selectionWithDescriptionData:**(NSData *)*newData*

Allocates an NSSelection instance and initializes it using a copy of the data indicated by *description* to describe the selection. As mentioned in the Class Description, the description should ideally be architecture-independent and should contain some sort of version information.

See also: **– initWithDescriptionData:**, **– descriptionData**

Instance Methods

descriptionData

– (NSData *)**descriptionData**

Returns the data that describes the selection as set by **selectionWithDescriptionData:** or **initWithDescriptionData:**.

initWithDescriptionData:

– (id)**initWithDescriptionData:**(NSData *)*description*

Initializes a newly allocated NSSelection instance using a copy of the data indicated by *description* to describe the selection. As mentioned in the Class Description, the description should ideally be architecture-independent and should contain some sort of version information. Returns **self**.

See also: **+ selectionWithDescriptionData:**, **– descriptionData**

initWithPasteboard:

– (id)**initWithPasteboard:**(NSPasteboard *)*pasteboard*

Initializes a newly allocated NSSelection instance from data on the specified pasteboard. If the NSSelection can't be initialized for any reason (for example, if data of type NSSelectionPboardType isn't found on the pasteboard) the new instance is freed and **nil** is returned. Returns the newly-initialized NSSelection instance (which may be other than **self**).

See also: – writeToPasteboard:

isWellKnownSelection

– (BOOL)**isWellKnownSelection**

Returns YES if the NSSelection is one of the three well-known selection types, and NO otherwise. There are well-known selection types for an entire document, the current selection, and for an empty selection.

See also: + **allSelection**, + **currentSelection**, + **emptySelection**

writeToPasteboard:

– (void)**writeToPasteboard:**(NSPasteboard *)*pasteboard*

Writes the NSSelection to the pasteboard *pasteboard*. A copy of the selection can then be retrieved by initializing a new NSSelection from the pasteboard using **initWithPasteboard:**.