InStore:(IXStore *)aStoreInitializes a new store client in aStore
FromBlock:(unsigned int)aHandleInitializes a store client from data previously stored in the
inStore:(IXStore *)aStoreblock identified by aHandle in aStore
eFromStoreRemoves the store client's storage from the IXStore and frees the run-time object
eeFromBlock:(unsigned int)aHandleFrees the data for the store client identified by aHandle in
inStore:(IXStore *)aStoreaStore, without necessarily creating an instance

Block:(unsigned int *)aHandleGets the identifier of the block owned by the store client,
andStore:(IXStore **)aStoreand the IXStore that the block exists in

Comparator:(IXComparator *)aComparatorSets the function used to compare items also
andContext:(const void *)aContextprovides aContext as arbitrary data to use in comparison
Comparator:(IXComparator **)aComparatorGets the comparator function and context
andContext:(const void **)aContext

ComparisonFormat:(const char *)formatSets the data format of items compared by the receiver
nst char *)comparisonFormatReturns the data format of items compared

OOL)setKey:(void *)aKeySets the position of the receiver to aKey, if it exists,
andLength:(unsigned int)aLengthotherwise to where aKey would logically be returns YES if aKey exists
OOL)getKey:(void **)aKeyGets the key for the receiver's position and its length,
andLength:(unsigned int *)aLengthsliding the receiver forward in the key space if needed and if possible returns YES if the
receiver ends up on a key

OOL)setFirstPositions the receiver at the first key if there is one returns YES if there is one, NO if not
OOL)setNextMoves the receiver forward one key value and returns YES if there's a key there

AttributeParsers:(List *)aListSets the IXAttributeParsers used to parse files
AttributeParsers:(List *)aListReturns in aList the IXAttributeParsers used to parse files

GeneratesDescriptions:(BOOL)flagSets whether descriptions are generated automatically for files indexed
OOL)generatesDescriptionsReturns whether descriptions are generated automatically for files indexed

UpdatesAutomatically:(BOOL)flagSets whether the file finder automatically updates its indexes upon finding
OOL)updatesAutomaticallyReturns whether the file finder automatically updates its indexes

CrossesDeviceChanges:(BOOL)flagSets whether the file finder indexes or searches files on a different device
OOL)crossesDeviceChangesReturns whether the file finder indexes or searches files on a different device fro
FollowsSymbolicLinks:(BOOL)flagSets whether the file finder follows symbolic links when building indexes
OOL)followsSymbolicLinksReturns whether the file finder follows symbolic links when building indexes
ScansForModifiedFiles:(BOOL)flagSets whether the file finder scans for files whose modification times have
OOL)scansForModifiedFilesReturns whether the file finder scans for modified files

IgnoredTypes:(const char *)typesSets to types the types of files that won't be indexed
ar *)ignoredTypesReturns the types of files that aren't indexed
IgnoredNames:(const char *)namesSets to names the literal, base names of files that won't be indexed
ar *)ignoredNamesReturns the names of files that aren't indexed

nst char *)rootPathReturns the base path for the file finder's index

ordManagerReturns the object that stores the file finder's IXFileRecords

PostingList *)performQuery:(const char *)aQuery
atPath:(const char *)pathEvaluates aQuery for sender returning in an
forSender:senderIXPostingList the IXFileRecords that match
pQueryForSender:senderStops the query requested by sender

Finder:(IXFileFinder *)aFinderAsynchronously notifies the sender of a
didFindFile:(IXFileRecord *)aRecordperformQuery:atPath:forSender: message that aRecord matches the

Finder:(IXFileFinder *)aFinderAsynchronously notifies the sender of an
didFindList:(IXPostingList *)aListperformQuery:atPath:forSender: message that the IXFileRecords in aLi

Finder:(IXFileFinder *)aFinderAsynchronously notifies the sender of an
willAddFile:(IXFileRecord *)aRecordupdateIndexAtPath:forSender: message that aRecord is about to be

signed int)getLexeme:(char *)aStringPuts the next lexeme from stream into aString
inLength:(unsigned int)aLength
fromStream:(NXStream *)stream

signed int)foldCase:(char *)aStringReduces aString to lowercase letters
inLength:(unsigned int)aLength

WithName:(const char *)aNameInitializes a new store client under aName in filename
inFile:(const char *)filename

FromName:(const char *)aNameInitializes a store client from data previously stored under
inFile:(const char *)filenameaName in filename if flag is YES, changes can be
forWriting:(BOOL)flagwritten back to the file

eFromStoreRemoves the store client's storage from the IXStoreFile and frees the run-time object

eeFromName:(const char *)aNameFrees the data for the store file client identified by aName in
andFile:(const char *)filenamefilename, without necessarily creating an instance

Name:(const char **)aNameGets the name of the store client, and the name of the file
andFile:(const char **)filenamethat the data exists in

signed int)addHandle:(unsigned int)aHandleAdds a postings to the set of postings
withWeight:(unsigned int)aWeight

noveHandle:(unsigned int)aHandleRemoves a postings from the set

signed int)countReturns the number of postings in the set

ptyEmpties all postings from the set

signed int)setHandle:(unsigned int)aHandleSets the selected posting to the one with aHandle and returns that
isn't in the set

signed int)getHandle:(unsigned int *)aHandleGets the handle and weight of the selected posting
andWeight:(unsigned int *)aWeight

signed int)setFirstHandleSets the selected posting to the first in the set and returns its handle, or 0 if there are

signed int)setNextHandleSets the selected posting to the next in the set and returns its handle, or 0 if there are

signed int)countReturns the number of records in the archive

dRecord:(unsigned int)aHandleReads the record identified by aHandle and returns the
fromZone:(NXZone *)zonecorresponding object allocated from zone

rce:aTranscriberNotifies the record identified by aHandle that it's been read
didReadRecord:(unsigned int)aHandle

rce:aTranscriberNotifies the record identified by aHandle that it's going to
willWriteRecord:(unsigned int)aHandlebe written

ishReadingAllows a record just read to reinitialize itself or provide a replacement