initWithName:inFile:

initFromName:inFile:forWriting:

freeFromStore

+ freeFromName:inFile:

Retrieving the block and store getName:andFile:

freeFromStore

Removes the client's data from its IXStoreFile's store file and frees the run-time object. A store fi
simply frees the run-time object without affecting any data in the store file. Returns nil.

free (Object)

getName:(const char **)aName andFile:(const char **)filename

Returns by reference the name of the client and of the store file in which it keeps its data. Also ret

A store file client must record at least its name (preferably in an instance variable). The file name
the IXStoreFile by sending it a filename message. If this is done, the implementation of this metho
values into aName and filename.

initFromName:(const char *)aName
inFile:(const char *)filename
forWriting:(BOOL)flag

Initializes the receiver from data previously stored in filename under entry aName. That data sho
by a previous invocation of the initWithName:inFile: method on the original instance of the store f
YES, filename is opened for reading and writing. If flag is NO, filename is opened for reading onl
to the store file's data, but they won't be flushed out to disk. The receiver isn't required to be of th
original creator of the store data, but it must be able to make sense of it. Returns self if successful,
can't be initialized (for example, if aName doesn't exist in filename, or if filename itself doesn't ex

One way to implement this method is to create an IXStoreDirectory, have it get the block for the en
initialize from that block. Here's a simple example, without any error handling:

Here's a simple example implementation, without any error handling:

initFromName:inFile:forWriting: