initInStore:
initFromBlock:inStore:
freeFromStore
+ freeFromBlock:inStore:
getBlock:andStore:

IXNameAndFileAccess initWithName:inFile:
initFromName:inFile:forWriting:
freeFromStore
+ freeFromName:andFile:
getName:andFile:

IXComparatorSetting setComparator:andContext:
getComparator:andContext:

IXComparisonSetting setComparisonFormat:
comparisonFormat

count

compact

Compacts the contents of the IXBTree so that it consumes as little space as possible (the average a
about 25%). This method may move significant amounts of data, so it can take some time to comp
will become slower for a while after this method is invoked, because data has to be moved around
Key search may become substantially faster however, since less data is paged from disk when sear
useful when occasional updates to an IXBTree are followed by lots of reading. Returns self.

optimizeForTime, optimizeForSpace

(unsigned int)count

Returns the number of key/value pairs stored in the IXBTree.

empty

Removes all key/value pairs from the IXBTree, and reduces its storage allocation to the smallest p
. Returns self.

(unsigned int)keyLimit

Returns the maximum allowable length of keys kept in the IXBTree. An IXBTreeCursor positions
the length of the key. That length should never be greater than the key limit. If it is, the IX_TooL
raised.

optimizeForSpace

Causes the IXBTree to move its contents around when inserting keys, so that the total amount of st
as small as possible. This will result in slower insertion times, but faster seek times. Your code sh
when the IXBTree will be used mostly for reading.

Once optimization is enabled, it can't be disabled however, the type of optimization may be switch
space (fast seek times) and time (fast insertions). Also, an IXBTree doesn't record its optimization
optimization must be re-enabled whenever an IXBTree is reconstituted.

optimizeForTime, compact

optimizeForTime

Causes the IXBTree to avoid moving its contents around when inserting keys, so that insertions ha
possible. This can result in more unused storage space and slower seek times. Your code should s
message when it will be making a lot of insertions.