Xindice 1.1 Commandline Tool Guide

$Revision: 1.9 $

by Jay Kalafus, Kimbro Staken

NOTICE:

Note:

If you notice incorrectness in this documentation, please notify Xindice community. Your feedback will help create better documentation.

1. Collection Management Commands

1.1. Adding a Collection

Adds a new collection to the database. When adding a collection under an existing collection hierarchy all parent collections must already exist in the database.

1.1.1. Summary:

```
xindice add_collection-c (or context) -n (or name) [-v (or ) ]
```

Action:	`add_collection`
Abberviated-Action:	ac
Access Level:	Admin

1.1.2. Parameters:

- `-c` The collection context under which to create the new collection.
- `-n` The name of the collection to create
- `-v` Show verbose output

1.1.3. Examples:

1.1.4. Creating a Top Level Collection

```
xindice add_collection -c /db -n pebbles
```

1.1.5. Creating a Sub Collection Under an Existing Collection

```
xindice add_collection -c /db/pebbles -n boulder
```

1.2. Deleting a Collection

Deletes a collection or subcollection from the database. If deleting a collection that has subcollections and documents they will also be deleted.

1.2.1. Summary:

`xindice delete_collection-c (or context) -n (or name) [-v (or ) ]`

Action:	`delete_collection`
Abberviated-Action:	`dc`
Access Level:	Admin

1.2.2. Parameters:

- `-c` The collection context under which to delete the collection.
- `-n` The name of the collection to delete.
- `-v` Show verbose output

1.2.3. Examples:

1.2.4. Deleting a Top Level Collection

```
xindice delete_collection -c /db -n pebbles
```

1.2.5. Deleting a Sub Collection

```
xindice delete_collection -c /db/pebbles -n boulder
```

1.3. List Collections

List all collections under the collection context given. If no collection is given then the root collection will be asumed.

1.3.1. Summary:

`xindice list_collections [-c (or context) ] [-v (or ) ]`

Action:	`list_collections`
Abberviated-Action:	`lc`
Access Level:	User

1.3.2. Parameters:

- `-c` The collection context under which all sub collections are listed
- `-v` Show verbose output

1.3.3. Examples:

1.3.4. Listing the Top Level Collections

```
xindice list_collections -c /db
```

1.3.5. Listing a Sub Collection

```
xindice list_collections -c /db/pebbles
```

2. Document Management Commands

2.1. Adding a Document

Adds a document to a collection or nested collection. Adding a document requires two parameters - the collection it will be stored under, and the file path to the document. If a document key is not provided an auto-generated system key will be used for the document. Documents cannot be added to collections that do not already exist. When entering the file path be sure to include the path and file extension.

2.1.1. Summary:

```
xindice add_document-c (or context) -f (or file path) [-n (or
key to assign to document) ] [-v (or ) ]
```

Action:	`add_document`
Abberviated-Action:	`ad`
Access Level:	User

2.1.2. Parameters:

- `-c` The collection context under which to add the document
- `-f` The complete file path to the document being added
- `-n` The key to assign to the document
- `-v` Show verbose output

2.1.3. Examples:

2.1.4. Adding a document to a collection with a key of "stones"

```
xindice add_document -c /db/pebbles -f /tmp/stones.xml -n stones
```

2.1.5. Adding a document to the root collection with an automatically generated key

```
xindice add_document -c /db -f /tmp/bookmarks.xml
```

2.2. Adding Multiple Documents

Adds multiple documents to a collection or a nested collection. This command requires two arguments: the collection to store the documents under and the directory path containing the documents. Documents can be added to collections as well as subcollections as long as they exist. Documents added will be assigned their file name as the document key. The optional "extension" parameter can be used to import documents with a certain file extension. Document keys are shown as they are created.

2.2.1. Summary:

```
xindice add_multiple_documents-c (or context) -f (or Directory
to use ) [-e (or file extenstion ) ] [-v (or ) ]
```

Action:	`add_multiple_documents`
Abberviated-Action:	`addmultiple`
Access Level:	User

2.2.2. Parameters:

- `-c` The collection context under which to add the documents
- `-f` The path to the directory to import documents from
- `-e` The file extension to use when importing documents
- `-v` Show verbose output

2.2.3. Examples:

2.2.4. Adding all files from a directory to a collection

```
xindice add_multiple_documents -c /db/pebbles -f /tmp/mydocs
```

2.2.5. Adding all files from a directory with an extension of ".xml"

```
xindice add_multiple_documents -c /db/pebbles -f /tmp/mydocs -e xml
```

2.3. Deleting a Document

Deletes an existing document from a collection or nested collection within the database.

2.3.1. Summary:

```
xindice delete_document-c (or context) -n (or document key) [-v (or ) ]
```

Action:	`delete_document`
Abberviated-Action:	`dd`
Access Level:	User

2.3.2. Parameters:

- `-c` The collection context under which to delete the document
- `-n` The key of the document to be deleted
- `-v` Show verbose output

2.3.3. Examples:

2.3.4. Deleting a Document from a collection

```
xindice delete_document -c /db/pebbles -n stones
```

2.4. Retrieving a Document

Retrieves an existing document from a collection or nested collection within the database. The complete path where the document will be stored is required. If the file path passed in does not exist, it will be created . If the file argument given already exists, it will be overwritten automatically.

2.4.1. Summary:

`xindice retrieve_document-c (or context) -n (or document key) -f (or file path and name ) [-v (or ) ]`

Action:	`retrieve_document`
Abberviated-Action:	`rd`
Access Level:	User

2.4.2. Parameters:

- `-c` The collection context under which to retrieve the document
- `-n` The key of the document to be retrieved
- `-f` The file path to store the document under
- `-v` Show verbose output
- `-u` The URI for the Xindice host to use. http://host:[port]

2.4.3. Examples:

2.4.4. Retrieving a document from a collection

```
xindice retrieve_document -c /db/pebbles -n stones -f /tmp/stones.xml
```

2.5. Importing a Directory Tree

Adds multiple documents into a collection or a nested collection within the database, copying

the directory structure into the database as collections. This command is designed to take a directory and create a collection from it's name. Directories under the directory will also be created as collections, duplicating the directory tree. The files within the specified directory are converted into Documents and then stored into the newly created collection with their filenames as the document keys. The optional "extension" parameter can be used to only import documents with a certain file extension. If the collection name used already exists it will be overwritten. Collection and document keys will are shown as they are created.

2.5.1. Summary:

`xindice import-c (or context) -f (or file path and name ) [-e (or file extenstion ) ] [-v (or ) ]`

Action:	`import`
Abberviated-Action:	`import`
Access Level:	Admin

2.5.2. Parameters:

- `-c` The collection context under which to import the documents
- `-f` The path to the documents being added
- `-e` The file extension to use when importing documents
- `-v` Show verbose output

2.5.3. Examples:

2.5.4. Importing all directories and documents

```
xindice import -c /db/pebbles -f /tmp/flintstones
```

2.5.5. Importing all directories and documents with an extension of ".xml"

```
xindice import -c /db/pebbles -f /tmp/flintstones -e xml
```

2.6. Exporting a Directory Tree

Creates an identical directory tree from a collection including any subcollections. The directory tree will be created starting at the directory passed in as an argument. This command requires the collection which you are eporting and the directory to export to as

arguments.

2.6.1. Summary:

```
xindice export-c (or context) -f (or directory path) [-v (or ) ]
```

Action:	`export`
Abberviated-Action:	`export`
Access Level:	User

2.6.2. Parameters:

- `-c` The collection context under which to export the document tree
- `-f` The directory to create the collection structure under
- `-v` Show verbose output

2.6.3. Examples:

2.6.4. Creating a directory tree indentical to a collection

```
xindice export -c /db/pebbles -f /tmp/pebbles
```

3. Collection Indexer Actions

3.1. Adding a Collection Indexer

Adds a new collection index or nested collection index to the database. In order to add the collection index, the collection it is being added to must already exist. This command allows you to supply optional parameters for setting the index page size, setting the maximum key size for the index, and the index data type.

3.1.1. Summary:

```
xindice add_indexer-c (or context) -n (or index name ) -p (or index pattern) [--pagesize (or pagesize) ] [--maxkeysize (or max key size) ] [-t (or index type) ] [-v (or ) ]
```

Action:	`add_indexer`
Abberviated-Action:	`ai`

Access Level:	Admin

3.1.2. Parameters:

- `-c` The collection context under which to add the indexer
- `-n` The name of the index being added
- `-p` The pattern used to create the index
- `--maxkeysize` The MaxKeySize for the index (default: 0 = none)
- `--pagesize` The PageSize for the index (default: 4096)
- `-t` The data type for the index to create. Possible values are: `string Non-normalized string value trimmed Normalized (whitespace stripped) string value short 16-bit signed short integer int 32-bit signed integer long 64-bit signed integer float 32-bit floating point value double 64-bit floating point value (XPath number) byte 8-bit signed byte char 16-bit signed character boolean 8-bit boolean value name Store document keys that contain the pattern`
- `-v` Show verbose output

3.1.3. Examples:

3.1.4. To create a collection index in the pebbles collection for docuemnts with a "rock" element

```
xindice add_indexer -c /db/pebbles -n rockindex -p rock
```

3.1.5. To create a collection index in the boulder sub collection for documents containing a rock elements with an attribute of "hard"

```
xindice add_indexer -c /db/pebbles/boulder -n hardindex -p rock@type
```

3.2. Deleting a Collection Indexer

Deletes a collection index or a nested collection index from the database. Deleting an index requires the collection and index name. If a collection index is deleted, the collection will still remain.

3.2.1. Summary:

```
xindice delete_indexer-c (or context) -n (or index name) [-v
```

```
(or ) ]
```

Action:	`delete_indexer`
Abberviated-Action:	`di`
Access Level:	Admin

3.2.2. Parameters:

- `-c` The collection context under which to delete the index
- `-n` The name of the indexer being deleted
- `-v` Show verbose output

3.2.3. Examples:

3.2.4. Delete a collecion index in the pebbles collection

```
xindice delete_indexer -c /db/pebbles -n hardindex
```

3.3. Listing Collection Indexers

Lists all collection indexers or nested collection indexes in the database. The listing of indexes works in a similar fashion to listing collections

3.3.1. Summary:

```
xindice list_indexers-c (or context) [-v (or ) ]
```

Action:	`list_indexers`
Abberviated-Action:	`li`
Access Level:	Admin

3.3.2. Parameters:

- `-c` The collection context under which to list the indexers
- `-v` Show verbose output

3.3.3. Examples:

3.3.4. To list all collection indexes from the pebbles collection

```
xindice list_indexers -c /db/pebbles
```

4. Xpath Query Actions

4.1. Executing an Xpath Query against a Collection

Execute an Xpath query against a collection or nested collection. The command requires two arguments : the collection to query against and the query string to use.

4.1.1. Summary:

`xindice xpath-c (or context) -q (or query) -s (or prefix=namespace) [-v (or ) ]`

Action:	`xpath`
Abberviated-Action:	`xpath`
Access Level:	User

4.1.2. Parameters:

- `-c` The collection context under which to execute the query
- `-q` The query to execute against the collection
- `-s` Semi-colon delimited list of namespaces for query in the form prefix=namespace-uri
- `-v` Show verbose output

4.1.3. Examples:

4.1.4. Run an Xpath query against all documents in the pebbles collection with "rock" node that has the type attribute = "hard"

```
xindice xpath -c /db/pebbles -q /rock[@type="hard"]
```

4.1.5. Run an Xpath query against all documents in the pebbles collection with "rock" node that has the type attribute = "hard" when the rock element is in the namespace http://www.bedrock.com

```
xindice xpath -c /db/pebbles -s "br=http://www.bedrock.com" -q /br:rock[@type="hard"]
```

5. Miscellaneous Actions

5.1. Shutting down the server

Shutdown the Xindice server

5.1.1. Summary:

`xindice shutdown [-v (or ) ]`

Action:	`shutdown`
Abberviated-Action:	`shutdown`
Access Level:	Admin

5.1.2. Parameters:

- `-v` Show verbose output

5.1.3. Examples:

5.1.4. Shutdown the server

```
xindice shutdown
```

5.2. Accessing online Help

Show the online help

5.2.1. Summary:

`xindice help [-v (or ) ]`

Action:	`help`
Abberviated-Action:	`help`
Access Level:	User

5.2.2. Parameters:

- `-v` Show verbose output

5.2.3. Examples:

5.2.4. Show online Help

```
xindice help
```