Dev Todo List

$Revision: 525711 $

by Vladimir R. Bossicard, Vadim Gritsenko

Table of contents

1. Introduction

For a new version of Xindice to be released, these following todos must be completed. Do not expect anything if these tasks are not done! So if you want to see a new release before the end of the millenium, choose your task!

Some plan items duplicating items from the main todo page.

2. Xindice 1.1 Release Plan

- **Admin Tool** Vet database copy/reindex tool (DatabaseRebuild), test on existing databases. Implement bat script for the tool.
- **Documentation** Create migration document from older releases to release 1.1 [VG].
- **Hash Filer** Replace HashFiler's hash function implementation with Java's hash function which gives better distribution. Such change to hash filer means that DatabaseRebuild tool should be used on any existing databases with hash filer backed collections. Once this change is done, hash filer can be un-deprecated [TB].
- ...

3. Xindice 1.2 Release Plan

- **Java** Revisit minimum JDK requirement.
- **Database Lock** Place a lock on database files using Java 1.4 APIs to prevent separate processes from working on the same database.
- **WebAdmin** Integrate new WebDAV capable WebAdmin into the main codebase, instead of ugly debug tool.
- **XML-RPC** Update to XML-RPC release 3.0 or newer.
- **XML-RPC** Extend Xindice XML-RPC API to optimize transmission of collection's symbol table between server and the client for reduced traffic and improved performance.
- **Caching** Revisit implementation of documents cache in the core Collection class. Clarify caching semantics, implement caching for all use cases (compressed, uncompressed, binary objects). Ensure dirty data can not be placed in the cache.
- **Admin Tool** DatabaseRebuild utility implements just one of low level database administration tasks. Implement tasks for checking database consistency, database recovery, and any other low level tasks and combine them all as a single `xindiceadmin` tool.
- **Paged** Introduce Paged interface to separate logical Hash and Tree structures from the physical storage mechanism. Implement nio based Paged as an option in addition to existing raf based Paged.
- **FullText** Implement full text indexing and searching. See scratchpad for some code.
- **Entity Catalogue** Xindice should have internal XML entity catalogue and utilize it.

Catalogue management operations should be made available via Collection API.
- **Meta Data** Review handling of creation, modification timestamps. This information should be primarily stored in the Filer's Record, and made available to the meta data service. This way a change to the document will require only single write to document collection, instead of two writes as currently is the case.
- **Inline Meta Data** Review handling of inline meta data. Currently it stores only `XML/Binary` bit, which can be made part of page header which already stores information such as creation and modification timestamp. This would also increase overall performance as data won't be copied by inline meta service implementation.
- **Configuration** It would be nice to implement user configurable default configuration parameters for all filers and collections, so that system or meta collections filers can be created with parameters specified by user instead of always using constants built into filer's code.
- ...

4. Xindice 1.3 Release Plan

- ...

5. Documentation

- Define the tables of content for the different guides and fill in the holes!