

Lucene2Whiteboard

Lucene 2.0

Release 2.0 will primarily make incompatible API improvements.

Release Migration Plan

Immediately prior to release 2.0 will be release 1.9, which will include all of the new 2.0 APIs, and all of the now-deprecated 1.x API's. 2.0 will be identical to 1.9, but with all deprecated APIs removed. To port an application to 2.0 one should first compile it against 1.9. Once it compiles against 1.9 without deprecation warnings, it should be 2.0 compatible.

API Changes

1. DONE: Replace Field factory methods (Field.Text, Field.Keyword, etc.) with a few methods that use type-safe enumerations, as described in: <http://www.mail-archive.com/lucene-user@jakarta.apache.org/msg08479.html>
2. DONE: Similarly, replace [BooleanQuery.add\(\)](#) with a type-safe enumeration, also as described in: <http://www.mail-archive.com/lucene-user@jakarta.apache.org/msg08479.html>
3. DONE: Replace public [IndexWriter](#) fields (mergeFactor, minMergeDocs, etc.) with get/set accessors. Also, minMergeDocs should be renamed maxBufferedDocs.
4. DONE: Rename [PhrasePrefixQuery](#) to be something like [MultiPhraseQuery](#). Also make [MultipleTermPositions](#) a private nested class of this, as this is the only place [MultipleTermPositions](#) is used.
5. DONE: Rename [InputStream](#) to [IndexInput](#) and [OutputStream](#) to [IndexOutput](#). Also add [BufferedIndexInput](#) and [BufferedIndexOutput](#) as the implementation used by FSDirectory, RAMDirectory, etc. This would permit unbuffered and native implementations (e.g., that use mmap) that could potentially speed things considerably.
6. DONE: Replace [DateField](#) with something that formats dates suitably for [RangeQuery](#).
7. DONE: Move language-specific analyzers into separate downloads. Also move analysis/de/WordlistLoader.java one level upwards, as it's not specific to German at all.
8. Remove public [PrintStream](#) infoStream from [IndexWriter](#). Instead use some kind of Logger which is customizable through the API to print debug, error and warning messages within lucene. The logger must not be an external library like log4j, it could be a small implementation directly in lucene to avoid references to external packages.
9. DONE: Add a non-static method isCurrent() to [IndexReader](#) and remove static getCurrentVersion() and lastModified methods: <http://www.mail-archive.com/lucene-dev@jakarta.apache.org/msg06143.html> (however, the deprecated methods will probably not be removed)
10. Implement an option for error handling described on the mailing list: [message](#) - if the [TooManyClauses](#) exception is kept, rename it to [TooManyClausesException](#).
11. (Hard) Make indexing more flexible, so that one could e.g., not store positions or even frequencies, or alternately, to store extra information with each position, or to even use different posting compression algorithms. This could be implemented by extending Field to specify a [FieldIndexer](#). A [FieldIndexer](#) would be passed each token and decide what about it to record, how to record it, etc. All fields with the same name must use the same [FieldIndexer](#) implementation. The [FieldIndexer](#) implementation would be serialized with the index. Detailed specification of a [FieldIndexer](#) API is required before this proposal can be seriously considered.
12. DONE: Modify [MultiFieldQueryParser](#) so that it behaves as most people expect: searching for A AND B in the fields body, title means that both terms must occur, but it doesn't matter whether they occur in title or body. The old behaviour must still be available by default so we stay compatible.
13. Deprecate [PorterStemFilter](#), in favor of the Snowball analyzers. This should also coincide with folding the Snowball codebase into the main Lucene CVS tree (to be built as a separate JAR but released with the main Lucene distributions).

Other Changes

Here's a list of planned changes that either don't affect the API or that can be implemented in an API compatible way:

1. Add support for span queries to query parser?
2. Implement a callback interface for processes which can run for several minutes like [IndexWriter.optimize\(\)](#). The idea is to define a simple public interface which can be implemented by developers using lucene. The object implementing the callback, could be passed to methods like [optimize\(\)](#) and can inform the caller when one of the steps to process has finished. This would make it much easier in interactive applications to inform the user that the system is working and not frozen.
3. Separate Query.toString from the actual conversion to a string representation. This will allow custom query parsing implementations to provide their own syntax easily. This could be done using an abstract factory that is looked up from each Query.toString implementation, with the current syntax being provided by [QueryParser](#) somehow.

Schedule

Lucene 2.0 was released 2006-05-26. Items not marked with "DONE" above did not make it into the release.