Item 57
git_comments:

1. for javadocs the data object for all different kind of field values pre-analyzed tokenStream for indexed fields
2. length/offset for all primitive types
3. * The TokesStream for this field to be used when indexing, or null. If null, the Reader value * or String value is analyzed to produce the indexed tokens.
4. * Expert: sets the token stream to be used for indexing and causes isIndexed() and isTokenized() to return true. * May be combined with stored values from stringValue() or binaryValue()
5. * Expert: change the value of this field. See setValue(String). * @deprecated use {@link #setTokenStream}
6. * The value of the field as a Reader, or null. If null, the String value or * binary value is used. Exactly one of stringValue(), * readerValue(), and getBinaryValue() must be set.
7. * The value of the field as a String, or null. If null, the Reader value or * binary value is used. Exactly one of stringValue(), * readerValue(), and getBinaryValue() must be set.
8. * True if the value of the field is to be indexed, so that it may be /** True if the value of the field should be tokenized as text prior to /** True if the term or terms used to index this field are stored as a term * True if terms are stored as term vector together with their offsets * True if terms are stored as term vector together with their token positions. /** True if the value of the field is stored as binary
9. * True if the value of the field is to be stored in the index for return with search hits.
10. * The value of the field in Binary, or null. * @see #stringValue()
11. * The TokenStream for this field to be used when indexing, or null. * @see #stringValue()
12. * The value of the field as a String, or null. * <p> * For indexing, if isStored()==true, the stringValue() will be used as the stored field value * unless isBinary()==true, in which case binaryValue() will be used. * * If isIndexed()==true and isTokenized()==false, this String value will be indexed as a single token. * If isIndexed()==true and isTokenized()==true, then tokenStreamValue() will be used to generate indexed tokens if not null, * else readerValue() will be used to generate indexed tokens if not null, else stringValue() will be used to generate tokens.
13. * The value of the field as a Reader, which can be used at index time to generate indexed tokens. * @see #stringValue()
14. test that the terms were indexed.
15. force segment flush so we can force a segment merge with doc3 later.
16. add 2 docs to test in-memory merging
17. force segment merge.

git_commits:

1. **summary:** LUCENE-1699: make Field.tokenStream usable with other stored field mechanisms
   message: LUCENE-1699: make Field.tokenStream usable with other stored field mechanisms git-svn-id: https://svn.apache.org/repos/asf/lucene/java/trunk@787437 13f79535-47bb-0310-9956-ffa450edef68

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** Field tokenStream should be usable with stored fields.
   description: Field.tokenStream should be usable for indexing even for stored values. Useful for many types of pre-analyzed values (text/numbers, etc)
   http://search.lucidimagination.com/search/document/902bad4eae20bdb8/field_tokenstreamvalue

jira_issues_comments:

1. incomplete patch for feedback - needs tests and further work to allow binary field values with token streams (I assume this would be desirable for numeric fields).
2. Another 2.9 huh? You own it then :)
3. Watch out - there's a new sheriff in town!
4. This group could develop hard for the rest of the year no problem by the looks. Someones got to herd these cats. Since we last semi decided it would be a good idea to release 2.9, we have rewritten half the core code base :)
5. **body:** Patch looks good: * Can you make sure CHANGES describes this new behavior (Field is allowed to have both a tokenStream and a String/Reader/binary value)? * The javadoc for readerValue is wrong (copy/paste from stringValue) * Can you spell out more clearly in the javadocs that even when a tokenStream value is set, one of String/Reader/binary may also be set, or, not, and if so, that "other" value is only used for stored fields. Eg, explain why one would use setTokenStream instead of setValue(TokenStream value).
   label: documentation
6. **body:** Patch attached. I've attempted to clean up some of the semantics of Fieldable - it's really a reasonable interface, with just a few redundancies (isBinary/getBinary). I've purposely avoided messing with Field more than necessary (adding yet more constructors, constants, etc). This change is back compatible as it just removes some unnecessary restrictions. I'll commit in a few days.
   label: code-design
7. committed.