TestIdeas

this is a page with ideas for how to improve lucene's testing infrastructure, a TODO or wishlist

List of ideas for improved test coverage

1. figure out some way to find 'uses default charset' bugs. These bugs exist, and unlike locale its not possible to just randomize in LuceneTestCase (once -Dfile.encoding is set on jvm init, its final). we might have to get creative here... (and ant is picky if you try to do something really evil like set it to some symbol/wingdings-like charset)
2. add a codecwrapper (used by MockRandomCodec) that wraps a terms dict and returns bytesrefs with non-zero offsets. Bugs probably exist that assume the BytesRef.offset=0, because thats what most codecs do.
3. when you call newField() in lucenetestcase, it should internally track with hashing (like RandomCodecProvider) and return consistent field properties for things like omitNorms/omitTF/use payloads in MockAnalyzer/... These things have special 'viral/antiviral' properties, so just doing random each time actually can result in worse test coverage, the only way to improve this is for LuceneTestCase to "remember".
4. now that xinclude works correctly, perhaps solr can, with sysprop substitution, have a 'random testing solrconfig' that e.g. sets indexwriter properties to random values (based on lucenetestcase's newIndexWriterConfig). This would give solr tests random numbers of segments, etc. Test configs could xinclude this thing to get good testing.
5. Jenkins should print all seeds when it runs; eg a test may pass but could run really slowly which could indicate a perf bug
6. JVM should with "dump heap on OOME"
7. ant test should try to create temp directories on different disks if multiple are available to find races if disk IO is really slow / really fast. Some testing machines have multiple disks from SSDs to old spinning ones.
8. RANDOM_MULTIPLIER should be a float not an int, and it should vary "around" the value (eg up to 20% bigger); we'd have to fix many places that use it to cast to int
9. Add @Weekly, and turn it on for Test2BTerms
10. Add getNumThreads() to LuceneTestCase that takes Runtime.getRuntime().availableProcessors() into account. Multi-threaded tests should use more threads than available processors to increase the chance to detect timing -sensitive data races so that at any given time some threads are running and some are switched out, thus reducing the predictability of interactions between threads.
11. REALTIME_BRANCH - Add Random DWPTThreadPool that randomly assigns ThreadStates to increase test coverage in DW if tests are single threaded.
12. Randomize FlushPolicy in RandomIndexWriter
13. Add a MockRandomMergeScheduler
14. IndexOutput should not write all bytes out on close() unless the file is fsynced unless there an exception occurs. that way we might be able to detect fsync problems earlier
15. Add ThrottledIndexInput just like ThrottledIndexOutput
16. RandomIndexWriter should randomly call MDW.crash() when writer is closed before opening reader
17. Move the randomly-insert-empty-IndexReaders from QueryUtil into LTC
18. LTC's AssertingIndexSearcher should sometimes impl search(...) by doing searches against the sub-searchers (segments, maybe coalescing some into single sub-searcher) and then use TopDocs.merge to merge the results
19. Add _TestUtil.randomBinaryTerm and use that from some tests, so we test that full binary terms are OK
20. Somehow, change up the system clock while tests are running (forwards and backwards) so we test any vulnerabilities because we rely on System.currentTimeMillis
21. Use a random codec that randomly uses different XXXFormat (Postings, StoredFields, TermVectors, SIS, etc.) 2. BaseTokenStreamTestCase should randomly throw an exc from the reader, and then make sure re-using the TS still produces the right tokens