Documentation Improvements

Documentation Improvements

A list of ideas (concrete or theoretical) on ways to improve documentation.

The origins of this page are this mailing list thread but people should feel free to add their own ideas...

- Documentation Improvements
- Website
- In Depth Docs on Key Concepts
- Tutorial
- Javadocs
- Wiki
- Misc Process Issues

Website

- Need more obvious info for for people just getting started, both on http://lucene.apache.org and http://lucene.apache.org/java ... most stuff is hidden in left nav
    - where/how to download
    - getting started tutorial
- Old news should be purged more often

In Depth Docs on Key Concepts

- There needs to be some docs that explain what analysis is at the top level, similar to the current Scoring documentation.
- Performance is another topic which would really benefit from a 'best practice' guide.

Tutorial

- The demo/tutorial needs to be brought into the current Lucene century.
- See LUCENE-805
- Most important part of this, I think is the "big picture" overview of why and when and how.

Javadocs

- automated tools to help identify missing javadocs?
- is there any way to easily allow annotation of javadocs (ala PHP and MySql)
- Need package level docs for every package- see:
    - index package - LUCENE-765
    - analysis package - LUCENE-925
    - document package - LUCENE-926
    - store package - LUCENE-927
- Need class level docs for every public class
- Need method level docs for every public method - particularly all methods used in any tutorial or "In Depth" doc (ie: scoring.html, and any similar docs that get written)
- "core plus contribs" nature of javadocs hard to understand for new users ... plethora of classes can be overwelming and hard to navigate - see: LUCENE-897
- better auditing of all javadocs in a class needs to be done when applying patches (docs elsewhere in the class may refer to things that have changed)

Wiki

- A best practices page on the Wiki would be great.
- Glossary of terms, etc.

Misc Process Issues

- Should we focus more on wiki docs or committed docs?
    - wiki is easier for community to contribute to
    - wiki pages can't be included in releases
- Before doing a release, we have 1-2 weeks of code freeze, and we focus on documentation and cleaning up bugs in JIRA.

- Get the Hudson JIRA integration stuff hooked in so we can know if patches are good faster, meaning we can turn around documentation patches, and others, faster
- How do we leverage vast amounts of info in mailing list archives?