Item 62
git_comments:

git_commits:

1. **summary:** LUCENE-4044: few steps closer to fixing compile
   message: LUCENE-4044: few steps closer to fixing compile git-svn-id: https://svn.apache.org/repos/asf/lucene/dev/branches/lucene2510@1364875 13f79535-47bb-0310-9956-ffa450edef68

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
   description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
2. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
   description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
3. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
   description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
4. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
   description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support

Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.

5. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
   description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
6. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
   description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
7. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
   description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
8. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
   description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
9. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
   description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
   label: code-design
10. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way

to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
label: code-design

11. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
    label: code-design
12. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
    label: code-design
13. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
    label: code-design
14. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
    label: code-design
15. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning

configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.

16. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
17. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
    label: code-design
18. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
19. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
    label: code-design
20. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
21. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way

to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.

22. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
23. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
24. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
25. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
26. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
27. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting

{{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.

28. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
29. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
    label: code-design
30. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
    label: code-design
31. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
32. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.

33. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
34. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
35. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
36. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
    label: test
37. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
38. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning

configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
label: code-design

39. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
40. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
41. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
42. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
43. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
44. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for

example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.

45. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
46. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
47. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
48. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
49. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
50. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
    description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way

to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.
label: code-design

51. **summary:** Add NamedSPILoader support to TokenizerFactory, TokenFilterFactory and CharFilterFactory
description: In LUCENE-2510 I want to move all the analysis factories out of Solr and into the directories with what they create. This is going to hamper Solr's existing strategy for supporting {{solr.*}} package names, where it replaces {{solr}} with various pre-defined package names. One way to tackle this is to use NamedSPILoader so we simply look up {{StandardTokenizerFactory}} for example, and find it wherever it is, as long as it is defined as a service. This is similar to how we support Codecs currently. As noted by Robert in LUCENE-2510, this would also have the benefit of meaning configurations could be less verbose, would aid in fully decoupling the analysis module from Solr, and make the analysis factories easier to interact with.

jira_issues_comments:

1. bq. This is going to hamper Solr's existing strategy for supporting solr.* package names Why is that? Why can't solr.WhitespaceTokenizerFactory also check the package that you're planning on moving WhitespaceTokenizerFactory to?
2. There will be alot of different packages, I assumed that cycling through them all would be undesirable.
3. With that said, I'm open to suggestions since I dont think this is going to do what I want it to do.
4. Hmm it seems that this process only supports singletons, which isn't much use to us.
5. why is it a problem? You just put the .class instance instead.
6. I'm not really sure how to do that, since the service has to implement NamedSPILoader.NamedSPI.
7. bulk cleanup of 4.0-ALPHA / 4.0 Jira versioning. all bulk edited issues have hoss20120711-bulk-40-change in a comment
8. I was wondering what happened to this issue, then i realized Chris was waiting on me to explain... here's a proof of concept patch (won't get by the generics policeman or anything, just showing that it works).
9. **body:** Chris: I'm gonna scratch out a branch here and see if we can flesh this out. At least I think we should give it a try as it would be pretty much a win for everyone to simplify this syntax and make these analysis modules real plugins (I'm frustrated with things like SOLR-3623 / the hassle people have to go thru to use some of the lucene analyzers) I expect Uwe to be horrified by what I did, but I'm hoping he will be willing to help it, and there is still a ton of work to get everything integrated :)
label: code-design
10. **body:** bq. it would be pretty much a win for everyone to simplify this syntax and make these analysis modules real plugins (I'm frustrated with things like SOLR-3623 / the hassle people have to go thru to use some of the lucene analyzers) Just to clarify: i believe you're saying that this would simplify things for end users because it would mean one less jar people need to load in (ie: bundle the FooTokenizerFactory in the same jar as the FooTokenizer so no need to a special solr contrib containing just the factory) ... correct? And in theory it would simplify things for developers because if we can remove things like contrib/solr/analysis-extras then things like SOLR-3664 are problematic in fewer situations. But (unless i'm missing something) nothing in this approach would actually eliminate the requirement that Solr users configure {{<lib ... />}} directives to load these external jars (and any third party dependencies) to get the factories+impls+third-party-deps ... correct?
label: code-design
11. **body:** {quote} Just to clarify: i believe you're saying that this would simplify things for end users because it would mean one less jar people need to load in (ie: bundle the FooTokenizerFactory in the same jar as the FooTokenizer so no need to a special solr contrib containing just the factory) ... correct? {quote} Yes. {quote} And in theory it would simplify things for developers because if we can remove things like contrib/solr/analysis-extras then things like SOLR-3664 are problematic in fewer situations. {quote} Yes. {quote} But (unless i'm missing something) nothing in this approach would actually eliminate the requirement that Solr users configure <lib ... /> directives to load these external jars (and any third party dependencies) to get the factories+impls+third-party-deps ... correct? {quote} There is no magic. they have to put things in their classpath. But the other major improvement i suggested was syntax too. Of course we must still support FQDN and also the solr.XXX: {noformat} <charFilter class="solr.HtmlStripCharFilterFactory"/> <tokenizer class="solr.StandardTokenizerFactory"/> <filter

class="solr.LowerCaseFilterFactory"/> <filter class="solr.PorterStemFilterFactory"/> {noformat} But I think we should fix the examples and support (as the "new" syntax): {noformat} <charFilter name="htmlstrip"/> <tokenizer name="standard"/> <filter name="lowercase"/> <filter name="porterstem"/> {noformat}
label: code-design

12. **body:** bq. There is no magic. they have to put things in their classpath. Just checking ... sometimes you and Uwe bust out stuff that is indistinguishable from magic. bq. But I think we should fix the examples and support (as the "new" syntax) Hmmm... yeah i'm -0 on that part ... i mainly just worry that it would be confusing for users if some places in the config (charfilter, tokneizer, tokenfilter, analyzer) supported these short names, but other places (fieldtype, requesthandler, serachcomponent, etc...) didn't and required a FQN ClassName or solr.ClassName --- FWIW: The other issue to consider here is how the packaging of these "uber-module-jars" will work in Solr binary pacakges ... right now it's the solr contribs that provide the factories that ensure the lucene modules they depend on make it into the final tgx/zip files ... if lucene modules start providing the factories themselves, we'll need some build.xml shenanigans to copy those module jars into the solr package dirs.
    label: code-design
13. **body:** {quote} Hmmm... yeah i'm -0 on that part ... i mainly just worry that it would be confusing for users if some places in the config (charfilter, tokneizer, tokenfilter, analyzer) supported these short names, but other places (fieldtype, requesthandler, serachcomponent, etc...) didn't and required a FQN ClassName or solr.ClassName {quote} Well solr can keep its verbose syntax then: I'll leave it alone. Ill just support backwards (class=solr.xxx) via the SPI. {quote} FWIW: The other issue to consider here is how the packaging of these "uber-module-jars" will work in Solr binary pacakges ... right now it's the solr contribs that provide the factories that ensure the lucene modules they depend on make it into the final tgx/zip files ... if lucene modules start providing the factories themselves, we'll need some build.xml shenanigans to copy those module jars into the solr package dirs. {quote} Its not anything we arent dealing with already. We already have shenanigans copying that stuff over. its just less jars.
    label: code-design
14. **body:** bq. Chris: I'm gonna scratch out a branch here and see if we can flesh this out. Thanks for coming back to this Robert! I was just thinking about returning to this issue. {quote} At least I think we should give it a try as it would be pretty much a win for everyone to simplify this syntax and make these analysis modules real plugins (I'm frustrated with things like SOLR-3623 / the hassle people have to go thru to use some of the lucene analyzers) {quote} Absolutely. I'm a huge fan of the simplified syntax. Every consumer of Lucene, Solr or not, would benefit from being able to easily retrieve these Factories. Looking over the patch I see what you meant and I really like it. I wonder whether we should integrate newInstance into NamedSPILoader and add getName() to the Factorys? The getName impl can then do the classname clipping like currently done in the patch.
    label: code-design
15. Chris: working on it :) my first goal: get compile and tests working again. Next: i would like TestAllFactoriesWorkViaSPI (we should be able to do that) And also: TestAllAnalysisComponentsHaveFactories (we can use TestRandomChains-like sweeping to find them) This would prevent anything from being out of sync.
16. Haha sweet! {quote} Next: i would like TestAllFactoriesWorkViaSPI (we should be able to do that) And also: TestAllAnalysisComponentsHaveFactories (we can use TestRandomChains-like sweeping to find them) This would prevent anything from being out of sync. {quote} Huge +1 to that.
17. **body:** {quote} Absolutely. I'm a huge fan of the simplified syntax. {quote} I think so too, but since some of this is contentious, lets just leave it for a future issue. Basically we can use the SPI to implement backwards compat, but in the future we can imagine more: * remove .java Analyzer files from lucene and replace with some declarative specification that uses the factories (e.g. json or whatever) * fix TestRandomChains or whatever to use SPI to find available tokenstreams, fix it to be a real integration test too. * allowing users to pick from available implementations in analysis gui to prototype up a chain and see what it looks like * simplified syntax as mentioned above Those were just my ideas. we can just leave these for future issues. On this issue we can make the pluggability simple, improve testing, make sure backwards is done right, and get it all past the generics policeman :)
    label: code-design
18. bq. I think so too, but since some of this is contentious, lets just leave it for a future issue. Yeah fair enough. bq. remove .java Analyzer files from lucene and replace with some declarative specification that uses the factories (e.g. json or whatever) This will be a great day.
19. **body:** {quote} bq. I think so too, but since some of this is contentious, lets just leave it for a future issue. Yeah fair enough. {quote} I wasn't objecting to supporting the shortenend syntax, i'm just worried that

using it in the examples might confuse people when the same shortened syntax doesn't work in other places. bq. Its not anything we arent dealing with already. We already have shenanigans copying that stuff over. its just less jars. it's something we're dealing with already, but the way we are dealing with it will probably need to change in some cases... right now (some) lucene-module jars get copied over into solr binary artifacts is because there are solr contribs that provide the factories and have build.xml files that copy the jars - if (some) solr contribs go away because the factories are included directly lucene module, then the solr contrib build.xml files will go away, and then we need new/differnet ways to ensure those lucene module jars get copied into solr binary packages ... correct?
label: code-design

20. {quote} right now (some) lucene-module jars get copied over into solr binary artifacts is because there are solr contribs that provide the factories and have build.xml files that copy the jars - if (some) solr contribs go away because the factories are included directly lucene module, then the solr contrib build.xml files will go away, and then we need new/differnet ways to ensure those lucene module jars get copied into solr binary packages ... correct? {quote} I can't make that contrib go away with this issue. It has a solr fieldtype as well for ICU. Because of that, for this issue we can just keep the status quo and not remove any contribs. Then there is nothing to deal with here :)
21. I was testing my Solr integration code and ran into needing to move MockTokenizerFactory (which is in solr/test-framework}. Moving it to lucene/test-framework alongside MockTokenizer doesn't seem to be an option since test-framework doesn't depend on the analysis module. Moving it into the analysis module main src doesn't seem to be an option either since test-framework is a test scoped dependency. It would be possible to put it into the analysis module test code but then it's not distributed like it is in test-framework. Any ideas?
22. Then the integration code has a problem? MockTokenizerFactory should work as is: its in the org.apache.solr package.
23. Hm, the problem is that MockTokenizerFactory isn't accessible via SPI. Should I add a META-INF TokenizerFactory SPI file in Solr? or are you suggesting we should use the existing behaviour when a Factory can be found traditionally, and only use SPI when it can't.
24. Chris male: you have it all backwards. {quote} or are you suggesting we should use the existing behaviour when a Factory can be found traditionally, and only use SPI when it can't. {quote} Try the SPI first... but then fall back totally on today's behavior. It *MUST* be this way or its a backwards break. I myself have made 3rd party analyzer projects and i put them in this package name on purpose so people could use the solr.XXX :)
25. -There is one problem with the SPI approach:- -SPI loads one instance per factory. The SPI loader returns only this instance. If you then set init paramters on it, you modify the only available singleton.- EDIT: You fixed that in the specail SPI loader :-) A second problem I see is the fact that on SPI init *all* factories are instantiated, but there is no way around that without another "Provider" interface inbetween. In all cases it is important that the factory does *not* initialize the class it creates, otherwise things like Kumoroji slow down. I hate the fact that SPILoader automatically instantiates the factory, but there is with Java 6 no way around, unless we write an own parser for META-INF files. Which would be a good option maybe.
26. {quote} A second problem I see is the fact that on SPI init all factories are instantiated, but there is no way around that without another "Provider" interface inbetween. In all cases it is important that the factory does not initialize the class it creates, otherwise things like Kumoroji slow down. I hate the fact that SPILoader automatically instantiates the factory, but there is with Java 6 no way around, unless we write an own parser for META-INF files. Which would be a good option maybe. {quote} This should only be a problem if Factories have heavy logic during construction, right? Most don't do anything until their init() method is called.
27. This seems like a big enough change that it should be targeted toward trunk and not 4x at this point.
28. {quote} This seems like a big enough change that it should be targeted toward trunk and not 4x at this point. {quote} Why? I Just started experimenting with this last night: you havent even given the issue time to be completed. I did my development in a branch to make it transparent: but its not mandatory to do that. I think its fine for this issue to be targeted at 4.x, in fact I think we are very close. Its no backwards break to the index format.
29. **body:** bq. Try the SPI first... but then fall back totally on today's behavior. It MUST be this way or its a backwards break. I myself have made 3rd party analyzer projects and i put them in this package name on purpose so people could use the solr.XXX Can we include this into the SPILoader itsself? So when it finds a name which looks like a full class name (but does not exist in the service list) it tries to Class.forName.asSubClass(clazz) it and instantiate it in newInstance(). The replacement of solr. to

something else should be maybe done in Solr as name-preprocessing. Maybe that's too much magic, just an idea.
label: code-design
30. **body:** bq. Can we include this into the SPILoader itsself? So when it finds a name which looks like a full class name (but does not exist in the service list) it tries to Class.forName.asSubClass(clazz) it and instantiate it in newInstance(). The replacement of solr. to something else should be maybe done in Solr as name-preprocessing. Maybe that's too much magic, just an idea. I'm experimenting with code that kind of does that. It does some pre-processing of the name before passing it of to the SPI Loader, but then it just falls back to existing behavior for simplicity (all inside the Solr side of the code).
label: code-design
31. bq. Why? Aside from other back compat risks, there are performance concerns. http://colabti.org/irclogger/irclogger_log/lucene-dev?date=2012-07-24
32. Like I said: I'm not even done yet. The issue just got started. I'm targeting 4.x: because I want the analysis modules to be successful in 4.0
33. I fully agree with Robert, we're working through the issues.
34. I did a review of the latest code and things are looking really good. What's still pending?
35. Yeah Uwe really went to town and rewrote everything (as I hoped!) I've just been adding more tests myself to shake anything out. I think this is ready to go in.
36. **body:** Yeah thanks for adding the great test coverage and improving the Factories. I'm +1 for this going in.
label: test
37. From my opinion, this is ready to go in! Real cool new tests. The only thing I want to add is an ANT task (e.g., using ASM like the other one), that creates the META-INF/services files automatically. We can then do the same in core for codecs or postings. It just collects all .class files from a FileSet that extends a specfic class and adds them to a services file. But that has time, currently all factories must be added manually, but this can go in! +1 to let this go in ASAP, as it outdates very fast (the many moves will make merging fast and changes to trunk's factories may get lost!
38. **body:** {quote} The only thing I want to add is an ANT task (e.g., using ASM like the other one), that creates the META-INF/services files automatically. We can then do the same in core for codecs or postings. It just collects all .class files from a FileSet that extends a specfic class and adds them to a services file. But that has time, currently all factories must be added manually, but this can go in! {quote} I agree this would be nice, for safety in the meantime I added TestAllAnalyzersHaveFactories, which checks that both we have a factory for every Tokenizer/TokenFilter/CharFilter and that the SPI is configured correctly (since it uses forName to validate this) and fails if we don't. Its not a perfect solution but just to prevent stupid mistakes.
label: code-design
39. For the provider-auto-config I found https://code.google.com/p/spi/, which is a javac annotation processor. Instead of adding the factories, we have to add a @ProviderFor(TokenizerFactor.class) on the implementation class (instead of adding it to the crazy file in resources folder). javac then automatically writes the META-INF file to the compiler output. As this annotation processor is an SPI by itsself and can be added to ANT's javac task as additional classpath (to find the tool), it will do this while compiling without any further configuration. For Eclipse you have to add annotation processing, but this can be enabled in ant eclipse. The cool thing is, that we dont need to make crazy Eclipse classpaths, because the META-INF/services folder in resources can be nuked and no file collisions can happen. I am still digging to have it fully automatic, but this approach is used by many projects.
40. Shall we merge and then address automation?
41. As I said, I will open separate issue! +1 to merge forward to branch and then reintegrate + commit! And merge back to 4.x (also asap, otherwise we might get out of sync, too because changes in factory classes in 4.x might get lost, too) I will now merge the branch up to trunk, reintegrate can come afterwards.
42. Here the monster patch for merging against trunk (caused by lot of SVN moves). I will try to get a simplier one which shows actual changes in non-moved classes only (explude additions/removals).
43. Short patch with no additions/deletions - just showing changes in already existing files. As you see changes in Solr and Lucene are minimal. Unfortunately it does not show all changes in the Lucene SPI utils (SPIClassIterator,...)
44. Thanks Uwe, patches look good, +1 to committing
45. One thing for later cleanup (I just add these points here to add further subtasks to parent issues once this is committed): Solr/contrib/analysis-extras is now almost obsolete and feels like just a JAR file duplicator - it only contains a standalone lib/ folder and one single class ICUCollationKeyField.java. We should nuke that and maybe find a good solution for the CollationKeyField (copy to core?, use Reflection, so it

only works with ICU when available,...). With the current approach you can simply copy the JAR files from the advanced analyzers in Lucene to your solr/lib plugin folder and they are available (depending on solrconfig.xml, of course). That's the most cool thing on the whole issue! No need to ship them with distribution, maybe add a downloader script to Solr that fetches the required JARS from maven/ivy and install them in you Solr instance's lib folder.

46. I will commit this to trunk and then we backport soon, because this gets outdated very fast.
47. Committed trunk revision: 1365586 I will remove branch now and start to backport.
48. Patches for 4.x (like trunk, one complete, one diffs only). It was a little bit of work, as some factories were different and created conflicts on removal. I also had to move factories for old deprecated analysis components including some utility methods. Thanks Robert for help! Some additional changes: - ArabicLetterTokenizerFactory - CJKTokenizerFactory - ChineseFilterFactory - PatternReplaceCharFilterFactory - FSTSynonym's backwards layer - ... I will commit this now.
49. Committed 4.x revision: 1365673