Item 242
git_comments:

git_commits:

1. **summary:** Dummy commit to clean stalled gh PRs
   message: Dummy commit to clean stalled gh PRs This closes: #513 (superseded)

github_issues:

github_issues_comments:

github_pulls:

1. **title:** PutHBaseJSON processor treats all values as Strings
   body: The operator will now inspect the node value to determine type and convert as such. Numeric integral - Long (assumes widest type) Numeric not integral - Double (assumes widest type) Logical - Boolean everything else (including current Complex Type logic) - String Values that represent the row key continue to be implictly treated as Strings by the processor

github_pulls_comments:

1. Ryan, thanks for submitting this PR! Is it possible that we could remove the dependency on hbase-client here? The reason being is that the hbase processor bundle purposely didn't include the hbase-client and instead used the hbase-client-service (which is where the hbase-client depndency is). This means that someone could implement an hbase-client-service for another version of hbase, such as 0.94 and the processors don't have to change at all. I think adding this dependency here would make that not possible because a specific version of hbase-client would now be bundled with the processors. Seems like the reason for the dependency was to use the Bytes class that provides the conversion, which makes total sense. I'm wondering if we could look at what the code is doing and possibly write our own util, or if there is some other utility library to do this maybe we can use that. Thoughts?
2. Yes, You're exactly right. I looked at the code in the Bytes util class and thought about doing that but then just imported the dependency. I expect that HBase users will use this Bytes class to decode the value when they interact with the data through their client application. If there's a change to the encoding/decoding scheme implemented in the Bytes class, our implementation would need to be updated to reflect this change. That was my only concern for including the logic rather than just leveraging the Bytes util.
3. Thats a great point about leveraging any changes they make to the Bytes class... I'm now thinking, what if we added simple methods to the HBaseClientService interface [1] that wrapped the calls to Bytes? Something like: ``` byte[] toBytes(boolean b) byte[] toBytes(long l) byte[] toBytes(double d) byte[] toBytes(String s) ``` Then HBase_1_1_2_ClientService would implement those methods and use the Bytes class from 1.1.2, and if someone implemented a 0.94 version it would be up to them to use the Bytes class from 0.94 or whatever they wanted to do. The processors never know they are dealing with the Bytes class. [1] https://github.com/apache/nifi/blob/master/nifi-nar-bundles/nifi-standard-services/nifi-hbase-client-service-api/src/main/java/org/apache/nifi/hbase/HBaseClientService.java
4. I like that idea.
5. Reviewing and going to merge into 0.x and master if all looks good...
6. @rtempleton please take a look at https://github.com/apache/nifi/pull/542 I've incorporated your work there with some additional changes which I explained in the PR description. Let me know if this seems good to you, and hopefully we should be able to get this into 0.7.
7. @rtempleton just merged this work as part of PR 542, thanks for the contribution! can we close this PR?
8. @rtempleton can you close this PR since it was included in PR 542? Thanks.

github_pulls_reviews:

jira_issues:

jira_issues_comments: