Review Checklist

Overview

Review Checklist defines a set of actions every reviewer must check before approving merge of a certain feature. Please make sure to follow these rules when submitting a patch. Otherwise it is likely to be rejected.

Requirements levels are identical to ones from RFC 2119 [1]:

1. **MUST** - This word mean that the definition is an absolute requirement of the specification.
2. **MUST NOT** - This phrase, or the phrase "SHALL NOT", mean that the definition is an absolute prohibition of the specification.
3. **SHOULD** - This word mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
4. **SHOULD NOT** - This phrase mean that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.

Checklist

1. API
    a. API compatibility **MUST** be maintained between minor releases. Do not remove existing API interfaces, classes, and methods or change their signatures, deprecate them instead
    b. Default behavior **SHOULD NOT** be changed between minor releases, unless absolutely needed. If a change is made, it **MUST** be described in "Migration Guide" [2]
    c. New operation **MUST** be well-documented in code (javadoc, dotnetdoc): documentation must contain method's purpose, description of parameters and how their values affect the outcome, description of return value and it's default, behavior in negative cases, interaction with other operations and components
    d. API parity between Java and .NET platforms **SHOULD** be maintained when operation makes sense on both platforms. If method cannot be implemented in a platform immediately, new JIRA ticket **MUST** be created and linked to current ticket
    e. API parity between thin clients (Java, .NET) **SHOULD** be maintained when operation makes sense on several clients. If method cannot be implemented in a client immediately, new JIRA ticket **MUST** be created and linked to current ticket
    f. All exceptions thrown to a user **SHOULD** have explanation how to resolve, workaround or debug an error
2. Compatibility
    a. Persistence backward compatibility **MUST** be maintained between minor releases. It should be possible to start newer version on data files created by the previous version
    b. Thin client forward and backward compatibility **SHOULD** be maintained between two consecutive minor releases. If compatibility cannot be maintained it **MUST** be described in "Migration Guide" [2]
    c. JDBC and ODBC forward and backward compatibility **SHOULD** be maintained between two consecutive minor releases. If compatibility cannot be maintained it **MUST** be described in "Migration Guide" [2]
3. Tests
    a. New functionality **MUST** be covered with unit tests for both positive and negative use cases
    b. Patch for a bug **SHOULD** have a test confirming that the bug is fixed
    c. All test suites **MUST** be run on TeamCity [3] before merge to master, there **MUST NOT** be any test failures. Not important test failures should be muted and handled according to [4] process.
4. Misc
    a. Code style **MUST** be followed as per Ignite's Coding Guidelines [5].
    b. Implementor **MUST** decide whether `Docs Requiered` flag should be left `ON`, reviewer **MUST** check whether decision was correct or not.

[1] https://www.ietf.org/rfc/rfc2119.txt

[2] https://github.com/apache/ignite/blob/master/MIGRATION_GUIDE.txt

[3] https://ci.ignite.apache.org/

[4] Make Teamcity Green Again

[5] Coding Guidelines