GitGuidelines

This is a proposal/draft how we should coordinate our work after migrating HttpCore and HttpClient to Git. Especially to avoid a messy log.

Typical Issue Workflow

1. Branch off a release branch (e.g., 4.4.x, 5.0.x) (`git checkout -b <release branch>/<JIRA id> master`) where `<JIRA id>` being the JIRA issue you have assigned to yourself, e.g., HTTPCORE-123 or HTTPCLIENT-689. Exmaple: `git checkout -b 4.4.x/HTTPCORE-123 4.4.x`.
2. Work on your issue and create as many commits as you want/need
3. Polish it, squash it or fix it up into a single commit
4. Ask for a review if you are uncertain

5. Take care of a proper commit message (good reads: 1 and 2): Put the title of the JIRA issue, e.g., [HTTPCORE-123] Memory leak in response, in the first line, followed by an explanation why you did take this approach. The ticket desc contains the issue, your commit message contains the solution. If in doubt, ask for help and give people a couple of days to react.

6. Request the release manager to merge your banch back to the release branch and make sure that this merge won't incur a merge commit
7. When you close the issue, put a link to your commit to create a direct relation between issue and solution.

Side Notes

1. Never rewrite (rebase) history on master or any other long-lived branch because you will break others. Only the release manager is entitled to clean up history upto 72 hours after a commit if it is absolutely necessary
2. If a change comes for a PR on GitHub:
    - Apply the same above rules
    - Don't steal authorship
    - Let the reporter polish his work
    - Amend the message at the end with "This closes/fixes #xy" and push.