Bylaws

1. Introduction

1.1.

- This document defines the bylaws under which the Apache jclouds project operates. It defines the roles and responsibilities of the project, who may vote, how voting works, how conflicts are resolved and specifies the rules for specific project actions.

1.2.

- jclouds is a project of the Apache Software Foundation. The foundation holds the trademark on the name "jclouds" and copyright on Apache code including the code in the jclouds codebase. The foundation FAQ explains the operation and background of the foundation.

1.3.

- jclouds operates under a set of principles known collectively as the "Apache Way". Those principles are: Transparancy, consensus, non-affiliation, respect for fellow developers, and meritocracy, in no specific order.

2. Roles and Responsibilities

- Apache projects define a set of roles with associated rights and responsibilities. These roles govern what tasks an individual may perform within the project. The roles are defined in the following sections:

2.1. Users

- The most important participants in the project are people who use our software. Users can contribute to the Apache projects by providing feedback to developers in the form of bug reports and feature suggestions. As well, users can participate in the Apache community by helping other users on mailing lists and user support forums. Users who participate in the project through any mechanism are considered to be Contributors.

2.2. Contributors

- Contributors are all of the volunteers who are contributing time, code, documentation, or resources to the jclouds Project. Contributions are not just code, but can be any combination of documentation, testing, user support, code, code reviews, bug reporting, community organizing, project marketing, or numerous other activities that help promote and improve the Apache jclouds project and community.

A Contributor that makes sustained, welcome contributions to the project may be invited to become a Committer by the PMC. The invitation will be at the discretion of a supporting PMC member.

2.3. Committers

- The project's Committers are responsible for the project's technical management. Committers have access to all project source control repositories. Committers may cast binding votes on any technical discussion regarding the project (or any sub-project).

2.3.1.

- Committer access is by invitation only and must be approved by a lazy consensus of the active PMC members.

2.3.2.

- All Apache Committers are required to have a signed Individual Contributor License Agreement (ICLA) on file with the Apache Software Foundation. There is a Committer FAQ which provides more details on the requirements for Committers at Apache.

2.3.3.

- A Committer who makes a sustained contribution to the project may be invited by the PMC to become a member of the PMC, after approval of the PMC.

2.4. Project Management Committee

- The Project Management Committee (PMC) for Apache jclouds is responsible to the board and the ASF for the management and oversight of the Apache jclouds codebase.

2.4.1.

- The responsibilities of the PMC include:

2.4.1.1.

- Fostering, supporting and growing the project's community.

2.4.1.2.

- Deciding what is distributed as products of the Apache jclouds project. In particular all releases must be approved by the PMC.

2.4.1.3.

- Maintaining the project's shared resources, including the codebase repository, mailing lists, websites.

2.4.1.4.

- Speaking on behalf of the project.

2.4.1.5.

- Resolving license disputes regarding products of the project.

2.4.1.6.

- Nominating new PMC members and committers.

2.4.1.7.

- Maintaining these bylaws and other guidelines of the project.

2.4.2.

- Membership of the PMC is by invitation only and must be approved by a lazy consensus of active PMC members.

2.4.3.

- A PMC member is considered "emeritus" by their own declaration. An emeritus member may request reinstatement to the PMC. Such reinstatement is subject to lazy consensus of the active PMC members.

2.4.4.

- "Active PMC members" are all non-emeritus PMC members.

2.4.5.

- The chair of the PMC is appointed by the ASF board. The chair is an office holder of the Apache Software Foundation (Vice President, Apache jclouds) and has primary responsibility to the board for the management of the projects within the scope of the jclouds PMC. The chair reports to the board quarterly on developments within the jclouds project. The chair must be an active PMC member.

2.4.6.

- If the current chair of the PMC resigns, or the term of the current chair expires, the PMC will attempt to reach consensus on a new chair through discussion, confirming that consensus via a vote to recommend a new chair using a lazy 2/3 majority voting method. In the case that consensus is not achieved, the PMC will vote for a chair using Single Transferable Vote (STV) voting. Due to the fact that the discussions are about specific individuals, this vote would be held on the jclouds-private mailing list. The decision must be ratified by the Apache board.

2.4.7.

- The role of PMC chair will have a one year term. The intention of this term is to allow for a rotation of the role amongst the PMC members. This intention does not prohibit the PMC from selecting the same chair to serve consecutive terms.

3. Decision Making

- This section defines how voting is performed, the types of approvals, and which types of decision require which type of approval.

3.1. Voting

3.1.1.

- Decisions regarding the project are made by votes on the primary project development mailing list (dev@jclouds.apache.org). Where necessary, PMC voting may take place on the private jclouds PMC mailing list. Votes are clearly indicated by subject line starting with [VOTE]. Votes may contain multiple items for approval and these should be clearly separated. Voting is carried out by replying to the vote mail.

3.1.2.

- Voting may take four flavors:

3.1.2.1.

- +1 "Yes," "Agree," or "the action should be performed." In general, this vote also indicates a willingness on the behalf of the voter to be approached for help in "making it happen".

3.1.2.2.

- +0 This vote indicates a willingness for the action under consideration to go ahead. The voter, however, will not take an active role in that process.

3.1.2.3.

- -0 This vote indicates that the voter does not, in general, agree with the proposed action but is not concerned enough to prevent the action going ahead.

3.1.2.4.

- -1 This is a negative vote. On issues where consensus is required, this vote counts as a veto if valid (see 3.3.2) and binding. It may also be appropriate for a -1 vote to include an alternative course of action.

3.1.3.

- All participants in the jclouds project are encouraged to show their agreement with or against a particular action by voting. For technical decisions, only the votes of committers are binding. Non-binding votes are still useful for those with binding votes to understand the perception of an action in the wider jclouds community. For PMC decisions, only the votes of PMC members are binding.

3.1.4.

- Voting can also be applied to changes made to the jclouds codebase. These typically take the form of a veto (-1) in reply to the commit message sent when the commit is made.

3.1.5.

- Non-binding -1 votes are not considered to be vetos for any decision.

3.2. Approvals

- There are three types of approvals that can be sought. Section 3.4 describes actions and types of approvals needed for each action.

3.2.1.

- Lazy Consensus - Lazy consensus requires 3 binding +1 votes and no binding -1 votes.

3.2.2.

- Lazy Majority - A lazy majority vote requires 3 binding +1 votes and more binding +1 votes than binding -1 votes.

3.2.3.

- Lazy 2/3 Majority - Lazy 2/3 majority votes requires at least 3 binding votes and twice as many binding +1 votes as binding -1 votes.

3.3. Vetoes

3.3.1.

- Vetoes are only possible in a lazy consensus vote.

3.3.2.

- A valid, binding veto cannot be overruled. If a veto is cast, it must be accompanied by a valid reason explaining the reasons for the veto. Vetoes with no explanation are void. The validity of a veto, if challenged, can be confirmed by anyone who has a binding vote. This does not necessarily signify agreement with the veto - merely that the veto is valid.

3.3.3.

- If you disagree with a valid veto, you must lobby the person casting the veto to withdraw their veto. If a veto is not withdrawn, any action that has been vetoed must be reversed in a timely manner.

3.4. Actions

- This section describes the various actions which are undertaken within the project, the roles that have the right to start a vote on the action, the corresponding approval required for that action and those who have binding votes over the action.

3.4.1. Technical Decisions

- Technical decisions should normally be made by the entire community using consensus gathering, and not through formal voting.   Technical decisions must be made on a project development mailing list.   During the consensus gathering process, technical decisions may be vetoed by any Committer with a valid reason.   If a formal vote is started for a technical decision, the vote will be held as a lazy consensus of committers.   A ny user, contributor, committer or PMC member can initiate a technical decision making process.

3.4.2. Release Plan

- Defines the timetable and work items for a release. The plan also nominates a Release Manager.   A lazy majority of committers is required for approval.   Any committer or PMC member may call a vote. The vote must occur on a project development mailing list.

3.4.3. Product Release

- When a release of one of the project's products is ready, a vote is required to accept the release as an official release of the project.   Lazy Majority of active PMC members is required for approval.   Any committer or PMC member may call a vote. The vote must occur on a project development mailing list.

3.4.4. Adoption of New Codebase

- When the codebase for an existing, released product is to be replaced with an alternative codebase. If such a vote fails to gain approval, the existing code base will continue.   This also covers the creation of new sub-projects within the project.   Lazy 2/3 majority of active PMC members. Any committer or PMC member may call a vote. The vote must occur on a project development mailing list.

3.4.5. New Committer

- When a new committer is proposed for the project.   Lazy consensus of active PMC members.   Any active PMC member may call a vote. The vote must occur on the PMC private mailing list.

3.4.6. New PMC Member

- When a committer is proposed for the PMC.   Lazy consensus of active PMC members.   Any active PMC member may call a vote. The vote must occur on the PMC private mailing list.

3.4.7. Committer Removal

- When removal of commit privileges is sought. Note: Such actions will also be referred to the ASF board by the PMC chair   Lazy 2/3 majority of active PMC members (excluding the committer in question if a member of the PMC).   Any active PMC member may call a vote. The vote must occur on the PMC private mailing list.

3.4.8. PMC Member Removal

- When removal of a PMC member is sought. Note: Such actions will also be referred to the ASF board by the PMC chair.   Lazy 2/3 majority of active PMC members (excluding the member in question)   Any active PMC member may call a vote. The vote must occur on the PMC private mailing list.

3.4.9. Modifying Bylaws

- Modifying this document.   Lazy majority of active PMC members   Any committer or PMC member may call a vote. The vote must occur on a project development mailing list.

3.5. Voting Timeframes

- Formal votes are open for a period of at least 72 hours, excluding weekends (Saturday and Sunday), to allow all active voters time to consider the vote.