Apache Pig Project Bylaws

Table of contents

1 Introduction

This document defines the bylaws under which the Apache Pig project operates. It defines the roles and responsibilities of the project, who may vote, how voting works, how conflicts are resolved, etc.

Pig is a project of the Apache Software Foundation The foundation holds the copyright on Apache code including the code in the Pig codebase. The foundation FAQ explains the operation and background of the foundation.

Pig is typical of Apache projects in that it operates under a set of principles, known collectively as the Apache Way. If you are new to Apache development, please refer to the Incubator project for more information on how Apache projects operate.

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. The majority of our contributors start out as users and guide their development efforts from the user's perspective.

Users contribute to the Apache projects by providing feedback to contributors in the form of bug reports and feature suggestions. As well, users participate in the Apache community by helping other users on mailing lists and user support forums.

2.2 Contributors

All of the volunteers who are contributing time, code, documentation, or resources to the Pig Project. A contributor that makes sustained, welcome contributions to the project may be invited to become a committer, though the exact timing of such invitations depends on many factors.

2.3 Committers

The project's committers are responsible for the project's technical management. Committers have access to a specified set of subproject's subversion repositories. Committers on subprojects may cast binding votes on any technical discussion regarding that subproject.

Committer access is by invitation only and must be approved by lazy consensus of the active PMC members. A Committer is considered emeritus by his or her own declaration or by not contributing in any form to the project for over six months. An emeritus committer may

request reinstatement of commit access from the PMC which will be sufficient to restore him or her to active committer status.

Commit access can be revoked by a unanimous vote of all the active PMC members (except the committer in question if he or she is also a PMC member).

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

A committer who makes a sustained contribution to the project may be invited to become a member of the PMC. The form of contribution is not limited to code. It can also include code review, helping out users on the mailing lists, documentation, etc.

2.4 Project Management Committee

The PMC is responsible to the board and the ASF for the management and oversight of the Apache Pig codebase. The responsibilities of the PMC include

- Deciding what is distributed as products of the Apache Pig project. In particular all releases must be approved by the PMC.
- Maintaining the project's shared resources, including the codebase repository, mailing lists, websites.
- Speaking on behalf of the project.
- Resolving license disputes regarding products of the project.
- Nominating new PMC members and committers.
- Maintaining these bylaws and other guidelines of the project.

Membership of the PMC is by invitation only and must be approved by a lazy consensus of active PMC members. A PMC member is considered emeritus by his or her own declaration or by not contributing in any form to the project for over six months. An emeritus member may request reinstatement to the PMC, which will be sufficient to restore him or her to active PMC member.

Membership of the PMC can be revoked by an unanimous vote of all the active PMC members other than the member in question.

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 Pig) and has primary responsibility to the board for the management of the projects within the scope of the Pig PMC. The chair reports to the board quarterly on developments within the Pig project.

The term of the chair is one year. When the current chair's term is up or if the chair resigns before the end of his or her term, the PMC votes to recommend a new chair using lazy consensus, but the decision must be ratified by the Apache board.

3 Decision Making

Within the Pig project, different types of decisions require different forms of approval. For example, the previous section describes several decisions which require 'lazy consensus' approval. This section defines how voting is performed, the types of approvals, and which types of decision require which type of approval.

3.1 Voting

Decisions regarding the project are made by votes on the primary project development mailing list user@pig.apache.org. Where necessary, PMC voting may take place on the private Pig PMC mailing list private@pig.apache.org. 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. Voting may take four flavours.

Vote	Meaning
+1	'Yes,' 'Agree,' or 'the action should be performed.' In general, this vote also indicates a willingness on the behalf of the voter in 'making it happen'.
+0	This vote indicates a willingness for the action under consideration to go ahead. The voter, however will not be able to help.
-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.
-1	This is a negative vote. On issues where consensus is required, this vote counts as a veto. All vetoes must contain an explanation of why the veto is appropriate. Vetoes with no explanation are void. It may also be appropriate for a -1 vote to include an alternative course of action.

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

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

that this should be a rare occurrence. All efforts should be made to discuss issues when they are still patches before the code is committed.

3.2 Approvals

These are the types of approvals that can be sought. Different actions require different types of approvals.

Approval Type	Definition
Consensus	For this to pass, all voters with binding votes must vote and there can be no binding vetoes (-1). Consensus votes are rarely required due to the impracticality of getting all eligible voters to cast a vote.
Lazy Consensus	Lazy consensus requires 3 binding +1 votes and no binding vetoes.
Lazy Majority	A lazy majority vote requires 3 binding +1 votes and more binding +1 votes that -1 votes.
Lazy Approval	An action with lazy approval is implicitly allowed unless a -1 vote is received, at which time, depending on the type of action, either lazy majority or lazy consensus approval must be obtained.
2/3 Majority	Some actions require a 2/3 majority of active committers or PMC members to pass. Such actions typically affect the foundation of the project (e.g. adopting a new codebase to replace an existing product). The higher threshold is designed to ensure such changes are strongly supported. To pass this vote requires at least 2/3 of binding vote holders to vote +1.

3.3 Vetoes

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. 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.

If you disagree with a valid veto, you must lobby the person casting the veto to withdraw his or her veto. If a veto is not withdrawn, the 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 corresponding approval required for that action and those who have binding votes over the action. It also specifies the minimum length of time that a vote must remain open, measured in business days. In general votes should not be called at times when it is known that interested members of the project will be unavailable.

Action	Description	Approval	Binding Votes	Minimum Length
Code Change	A change made to a codebase of the project and committed by a committer. This includes source code, documentation, website content, etc.	Lazy approval (not counting the vote of the contributor), moving to lazy majority if a -1 is received	Active committers	1
Release Plan	Defines the timetable and actions for a release. The plan also nominates a Release Manager.	Lazy majority	Active committers	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	Active PMC members	3
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	2/3 majority	Active PMC members	6

Action	Description	Approval	Binding Votes	Minimum Length
	creation of new sub-projects within the project.			
New Committer	When a new committer is proposed for the project.	Lazy consensus	Active PMC members	3
New PMC Member	When a committer is proposed for the PMC.	Lazy consensus	Active PMC members	3
Committer Removal	When removal of commit privileges is sought. Note: Such actions will also be referred to the ASF board by the PMC chair.	Consensus	Active PMC members (excluding the committer in question if a member of the PMC).	6
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.	Consensus	Active PMC members (excluding the member in question).	6
Modifying Bylaws	Modifying this document.	2/3 majority	Active PMC members	6