

Apache Voting Process

by The Apache Incubator Project

NOTICE: この文書は現在起稿中です....

Apacheにおける投票についての説明

1. 投票を通じたコンセンサスの測定

Apacheフレームワーク内で物事を達成するにあたっての基本的な側面の一つに「物事をコンセンサスをもって進めていく」というのがありますから、明らかに、どのようにそれが為されていくのかについての説明を行う必要があります。これは、「投票行為」によって行われます。

本質的に、3種類の投票があります：

1. コードの修正
2. パッケージのリリース
3. 手続き上のもの

手続き上の議題に関する投票については、他の方法の指示が予め無い限り、通常の形式つまり「賛成多数」方式が用いられます。つまり、もし「賛成票」が「反対票」よりも多い場合、その議題は「通過」したことになります。--これは、各々のカテゴリ内の投票数によりません（もし、コミュニティのコンセンサスを代表させるに及ばないほど少ない投票数であれば、その議題は一般的に続行されません。但し、修正ファクタとしての[lazy consensus](#)の説明書きをご覧ください）

コードの修正に関する投票については異なるモデルの手法を取ります。「反対票」が[veto](#)を構成するシナリオでは、覆される事はありません。再度、このモデルが「[lazy consensus](#)です」という宣言をもって投票にかけられた場合、「反対票」によ

る「終止符」の法則が覆る事はありません。通常 (lazy consensusではない) の場合、3つの賛成票と0の反対票が、「通過」の条件となります；もし必須の数のサポートを獲得するのに失敗すれば、通過しません。--典型的には、撤回か修正もしくは誰かが削除するまで開かれた議題として単に放置されるのみとなります。

パッケージのリリースが準備されたかあるいは未だか、に関する投票については、これまた異なるメカニズムを使います：リリースに賛成する3つのバインディング投票が少なくともあるでしょうか？これに関する詳細については、[以下](#)をご覧ください。

1.1. Binding Votes

Who is permitted to vote is, to some extent, a community-specific thing. However, the basic rule is that committers have binding votes, and all others are either discouraged from voting (to keep the noise down) or else have their votes considered of an indicative or advisory nature only.

That's the general rule. In actual fact, things tend to be a little looser, and procedural votes from non-committers are sometimes considered binding if the voter has acquired enough merit and respect in the community. Only votes by committers are considered binding on code-modification issues, however.

1.2. Implications of Voting

In some cases and communities, the exercise of a vote carries some responsibilities that may not be immediately obvious. For example, in some cases a favourable vote carries the implied message 'I approve and I'm willing to help.' Also, an unfavourable vote may imply that 'I disapprove, but I have an alternative and will help with that alternative.'

The tacit implications of voting should be spelt out in the community's guidelines. However, in no case may someone's vote be considered invalid if the implied commitment doesn't appear to be met; a vote is a formal expression of opinion, not of commitment.

If the [R-T-C](#) policy is in effect, a positive vote carries the very strong implied message, 'I have tested this patch myself, and found it good.' Similarly, a negative vote usually means that the patch was tested and

Apache Voting Process

found to be not-good, although the veto (for such it is in this case) may be based on other technical grounds.

1.3. Expressing Votes: +1, 0, -1, and Fractions

The voting process in Apache may seem more than a little weird if you've never encountered it before. Votes are represented as numbers between -1 and +1, with '-1' meaning 'no' and '+1' meaning 'yes.'

The in-between values are indicative of how strongly the voting individual feels. Here are some examples of fractional votes and ways in which they might be intended and interpreted:

+0: 'I don't feel strongly about it, but I'm okey with this.'

-0: 'I won't get in the way, but I'd rather we didn't do this.'

-0.5: 'I don't like this idea, but I can't find any rational justification for my feelings.'

++1: 'Wow! I like this! Let's do it!'

-0.9: 'I really don't like this, but I'm not going to stand in the way if everyone else wants to go ahead with it.'

+0.9: 'This is a cool idea and i like it, but I don't have time/the skills necessary to help out.'

Votes should generally be permitted to run for at least 72 hours to provide an opportunity for all concerned persons to participate regardless of their geographic locations.

1.3.1. Votes on Code Modification

For code-modification votes, +1 votes are in favour of the proposal, but -1 votes are [vetos](#) and kill the proposal dead until all vetoers withdraw their -1 votes.

Unless a vote has been declared as using [lazy consensus](#), three +1 votes are required for a code-modification proposal to pass.

Whole numbers are recommended for this type of vote, as the opinion being expressed is Boolean: 'I approve/do not approve of this change.'

1.3.2. Procedural Votes or Opinion Polls

TBS

1.3.3. Votes on Package Releases

Votes on whether a package is ready to be released follow a format similar to [majority approval](#) -- except that the decision is officially determined solely by whether at least three +1 votes were registered. Releases may not be vetoed. Generally the community will table the vote to release if anyone identifies serious problems, but in most cases the ultimate decision, once three or more positive votes have been garnered, lies with the individual serving as release manager. The specifics of the process may vary from project to project, but the 'minimum of three +1 votes' rule is universal.

1.4. Vetos

A code-modification proposal may be stopped dead in its tracks by a -1 vote by a qualified voter. This constitutes a veto, and it cannot be overruled nor overridden by anyone. Vetos stand until and unless withdrawn by their casters.

To prevent vetos from being used capriciously, they must be accompanied by a technical justification showing why the change is bad (opens a security exposure, negatively affects performance, etc.). A veto without a justification is invalid and has no weight.

2. Consensus Gauging through Silence

An alternative to voting that is sometimes used to measure the acceptability of something is the concept of [lazy consensus](#).

Lazy consensus is simply an announcement of 'silence gives assent.' When someone wants to determine the sense of the community this way, it might do so with a mail message such as:

"The patch below fixes bug #8271847; if no-one objects within three days, I'll assume lazy consensus and commit it."

Lazy consensus cannot be applied to code changes when the [review-then-commit](#) policy is in effect.