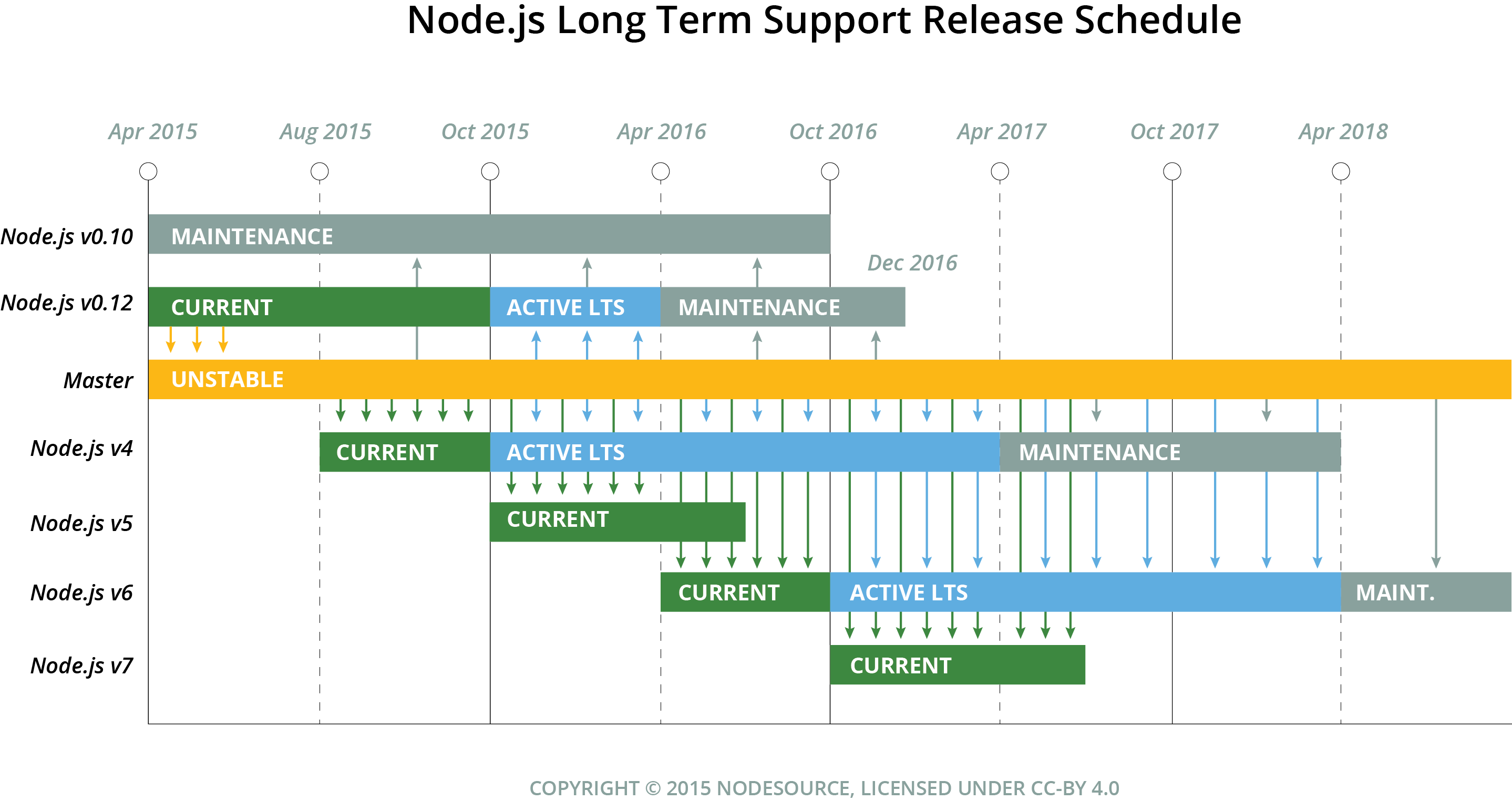
**Node.js Long-term Support Working Group**

**LTS schedule**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **LTS Status** | **Release** | **LTS Start** | **Maintenance Start** | **LTS End** |
| **Maintenance** | v0.10 | - | 2015-10-01 | 2016-10-01 |
| **Maintenance** | v0.12 | - | 2016-04-01 | 2016-12-31 |
| **Active** | v4 | 2015-10-01 | 2017-04-01 | 2018-04-01 |
| N/A | v5 | N/A | | |
| **Pending** | v6 | 2016-10-01 | 2018-04-01 | 2019-04-01 |

[](https://github.com/nodejs/LTS/blob/master/schedule.png)

**LTS Plan**

New semver-major releases of Node.js are cut from master every six months. New even-numbered versions (e.g. v6, v8, v10, etc) are cut in April. New odd-numbered versions (e.g. v5, v7, v9) are cut in October.

When a new *odd-numbered* major release is cut, the previous *even-numbered* major version transitions to the Long Term Support plan.

Every major version covered by the LTS plan will be actively maintained for a period of 18 months from the date it enters LTS coverage. Following those 18 months of active support, the major version will transition into "maintenance" mode for 12 additional months.

Given this schedule, there will be no more than two active LTS releases at any given time, overlapping for a maximum period of six months.

Once a major version enters LTS coverage, new features (semver-minor) may only be landed with consent of the CTC and the LTS Working Group. No semver-major changes other than those required for critical security fixes may be landed.

Changes in an LTS-covered major version are limited to:

1. Bug fixes;
2. Security updates;
3. Non-semver-major npm updates;
4. Relevant documentation updates;
5. Certain performance improvements where the risk of breaking existing applications is minimal;
6. Changes that introduce large amount of code churn where the risk of breaking existing applications is low and where the change in question may significantly ease the ability to backport future changes due to the reduction in diff noise.

Once a release moves into Maintenance mode, only ***critical*** bugs, ***critical*** security fixes, and documentation updates will be permitted.

Note that while it is possible that critical security and bug fixes may lead to *semver-major* changes landing within an LTS stream, such situations will be rare and will land as *semver-minor* bumps in the LTS covered version.

All LTS releases will be assigned a "codename" drawn from the names of elements on the Periodic Table of Elements. For each upcoming LTS release, the LTS Working Group will select a handful of candidate names and submit those for a collaborator vote.

**LTS Staging Branches**

Every LTS major version has two branches in the GitHub repository: a release branch and a staging branch. The release branch is used to cut new releases. Only members of the release team should land commits into the release branch. The staging branch is used to land cherry-picked or backported commits from master that need to be included in a future release.

For example, for Node.js v4, there is a v4.x branch and a v4.x-staging branch. When commits land in master that must be cherry-picked for a future Node.js v4 release, those must be landed into the v4.x-staging branch. When commits are backported for a future Node.js v4 release, those must come in the form of pull requests opened against the v4.x-staging branch. **Commits are only landed in the v4.x branch when a new v4.x release is being prepared.**

**Node abstraction layer**

It should be stated that the abstraction layer (currently NAN) should support all *current* LTS releases. Given that Active LTS will overlap for a period of 6 months, this means that the abstraction layer will, at any given point in time, fully support a maximum of 2 LTS releases.