Angular versioning and releases

We recognize that you need stability from the Angular framework. Stability ensures that reusable components and libraries, tutorials, tools, and learned practices don't become obsolete unexpectedly. Stability is essential for the ecosystem around Angular to thrive.

We also share with you the need for Angular to keep evolving. We strive to ensure that the foundation on top of which you are building is continuously improving and enabling you to stay up-to-date with the rest of the web ecosystem and your user needs.

This document contains the practices that we follow to provide you with a leading-edge application development platform, balanced with stability. We strive to ensure that future changes are always introduced in a predictable way. We want everyone who depends on Angular to know when and how new features are added, and to be well-prepared when obsolete ones are removed.

The practices described in this document apply to Angular 2.0 and later. If you are currently using AngularJS, see Upgrading from AngularJS. *AngularJS* is the name for all v1.x versions of Angular.

{@a versioning}

Angular versioning

Angular version numbers indicate the level of changes that are introduced by the release. This use of semantic versioning helps you understand the potential impact of updating to a new version.

Angular version numbers have three parts: `major.minor.patch`. For example, version 7.2.11 indicates major version 7, minor version 2, and patch level 11.

The version number is incremented based on the level of change included in the release.

- **Major releases** contain significant new features, some but minimal developer assistance is expected during the update. When updating to a new major release, you might need to run update scripts, refactor code, run additional tests, and learn new APIs.
- **Minor releases** contain new smaller features. Minor releases are fully backward-compatible; no developer assistance is expected during update, but you can optionally modify your applications and libraries to begin using new APIs, features, and capabilities that were added in the release. We update peer dependencies in minor versions by expanding the supported versions, but we do not require projects to update these dependencies.
- **Patch releases** are low risk, bug fix releases. No developer assistance is expected during update.

Note: As of Angular version 7, the major versions of Angular core and the CLI are aligned. This means that in order to use the CLI as you develop an Angular app, the version of `@angular/core` and the CLI need to be the same.

{@a updating}

Supported update paths

You can `ng update` to any version of Angular, provided that the following criteria are met:

- The version you want to update *to* is supported.
- The version you want to update *from* is within one major version of the version you want to upgrade to.

For example, you can update from version 11 to version 12, provided that version 12 is still supported. If you want to update across multiple major versions, perform each update one major version at a time. For example, to update from version 10 to version 12:

1. Update from version 10 to version 11.
2. Update from version 11 to version 12.

See Keeping Up-to-Date for more information about updating your Angular projects to the most recent version.

{@a previews}

Preview releases

We let you preview what's coming by providing "Next" and Release Candidates (`rc`) pre-releases for each major and minor release:

- **Next:** The release that is under active development and testing. The next release is indicated by a release tag appended with the `-next` identifier, such as `8.1.0-next.0` .
- **Release candidate:** A release that is feature complete and in final testing. A release candidate is indicated by a release tag appended with the `-rc` identifier, such as version `8.1.0-rc.0` .

The latest `next` or `rc` pre-release version of the documentation is available at next.angular.io.

{@a frequency}

Release frequency

We work toward a regular schedule of releases, so that you can plan and coordinate your updates with the continuing evolution of Angular.

Dates are offered as general guidance and are subject to change.

In general, expect the following release cycle:

- A major release every 6 months
- 1-3 minor releases for each major release
- A patch release and pre-release (`next` or `rc`) build almost every week

This cadence of releases gives eager developers access to new features as soon as they are fully developed and pass through our code review and integration testing processes, while maintaining the stability and reliability of the platform for production users that prefer to receive features after they have been validated by Google and other developers that use the pre-release builds.

{@a lts} {@a support}

Support policy and schedule

Dates are offered as general guidance and are subject to change.

All major releases are typically supported for 18 months.

- 6 months of *active support*, during which regularly-scheduled updates and patches are released.

- 12 months of *long-term support (LTS)*, during which only critical fixes and security patches are released.

The following table provides the status for Angular versions under support.

Version	Status	Released	Active Ends	LTS Ends
^13.0.0	Active	Nov 04, 2021	May 04, 2022	May 04, 2023
^12.0.0	LTS	May 12, 2021	Nov 12, 2021	Nov 12, 2022
^11.0.0	LTS	Nov 11, 2020	May 11, 2021	May 11, 2022

Angular versions v2 to v10 are no longer under support.

LTS fixes

As a general rule, a fix is considered for an LTS version if it resolves one of:

- a newly identified security vulnerability,
- a regression, since the start of LTS, caused by a 3rd party change, such as a new browser version.

{@a deprecation}

Deprecation practices

Sometimes "breaking changes", such as the removal of support for select APIs and features, are necessary to innovate and stay current with new best practices, changing dependencies, or changes in the (web) platform itself.

To make these transitions as straightforward as possible, we make these commitments to you:

- We work hard to minimize the number of breaking changes and to provide migration tools when possible.
- We follow the deprecation policy described here, so you have time to update your applications to the latest APIs and best practices.

To help ensure that you have sufficient time and a clear path to update, this is our deprecation policy:

- **Announcement:** We announce deprecated APIs and features in the change log. Deprecated APIs appear in the documentation with ~~strikethrough.~~ When we announce a deprecation, we also announce a recommended update path. For convenience, Deprecations contains a summary of deprecated APIs and features.
- **Deprecation period:** When an API or a feature is deprecated, it is still present in the next two major releases. After that, deprecated APIs and features are candidates for removal. A deprecation can be announced in any release, but the removal of a deprecated API or feature happens only in major release. Until a deprecated API or feature is removed, it is maintained according to the LTS support policy, meaning that only critical and security issues are fixed.
- **npm dependencies:** We only make npm dependency updates that require changes to your applications in a major release. In minor releases, we update peer dependencies by expanding the supported versions, but we do not require projects to update these dependencies until a future major version. This means that during minor Angular releases, npm dependency updates within Angular applications and libraries are optional.

{@a public-api}

Public API surface

Angular is a collection of many packages, sub-projects, and tools. To prevent accidental use of private APIs—and so that you can clearly understand what is covered by the practices described here—we document what is and is not considered our public API surface. For details, see Supported Public API Surface of Angular.

Any changes to the public API surface are done using the versioning, support, and depreciation policies previously described.

{@a labs}

Angular Labs

Angular Labs is an initiative to cultivate new features and iterate on them quickly. Angular Labs provides a safe place for exploration and experimentation by the Angular team.

Angular Labs projects are not ready for production use, and no commitment is made to bring them to production. The policies and practices that are described in this document do not apply to Angular Labs projects.