



Package versioning

Contents

Update for Debian package	2
Update for Apertis native packages	2
Backport of Debian update for Debian package	3
Backport of Apertis update for Debian package	3
Reference	4

Apertis is an source distribution based on Debian. As such it follows the same rules as Debian for the versioning of the packages and applications.

However, besides the general guidelines from upstream, it is useful to clarify different use cases in order to provide clear policies.

For this reason, the current document describe different scenarios and how to deal with them.

Update for Debian package

This is the most common case, and the simplest one. Apertis adds the suffix `+apertis` and an Apertis revision starting form `0`. Apertis specific changes bumps the Apertis revision.

Example:

Starting point:

- Debian: `dash 0.5.12-2`
- Apertis: `dash 0.5.12-2+apertis0 (v2025pre)`

After update:

- Apertis: `dash 0.5.12-2+apertis1 (v2025pre)`

Update for Apertis native packages

In this case, the upstream repo for the project is in Apertis, so a versioning scheme should be used which aligns with the release cycle and makes it easier to plan backports. Taking this into account Apertis uses a follow the `major.minor.patch` schema with `major=0`, `minor=release` and `patch=version`. This election aims to make it easier to backports changes and diverge to adapt to different Apertis release.

In case that the update is planned to be backported the version should only increment the patch number, even in the case of new release.

Example

Starting point:

- Apertis `apertis-dev-tools 0.2024.15 (v2025pre)`
- Apertis `apertis-dev-tools 0.2024.15 (v2024)`

36 After update that is planned to be backported:

- 37 • Apertis apertis-dev-tools 0.2024.16 (v2025pre)
- 38 • Apertis apertis-dev-tools 0.2024.16 (v2024)

39 However, in case of update not planned to be backported, the minor number
40 should increase and match the release number.

41 **Example**

42 Starting point:

- 43 • Apertis apertis-dev-tools 0.2024.15 (v2025pre)
- 44 • Apertis apertis-dev-tools 0.2024.15 (v2024)

45 After update that is **not** planned to be backported:

- 46 • Apertis apertis-dev-tools 0.2025.0 (v2025pre)
- 47 • Apertis apertis-dev-tools 0.2024.15 (v2024)

48 **Backport of Debian update for Debian package**

49 Due to the different release cycles between Debian and Apertis it is possible
50 that the same version of the package is found in multiple Apertis release. In
51 this case the backport is straight forward as the exact same version is applied
52 in older releases.

53 However, due to the needs of each Apertis release, in some occasions, the back-
54 port is not straight forward as the same Debian version of the package requires
55 different version in different Apertis releases. In this case a Apertis release suffix
56 should be added.

57 **Example:**

58 Starting point:

- 59 • Debian: dash 0.5.12-1
- 60 • Apertis: dash 0.5.12-1+apertis1 (v2025pre)
- 61 • Apertis: dash 0.5.12-1+apertis0 (v2024)

62 After update:

- 63 • Debian: dash 0.5.12-2
- 64 • Apertis: dash 0.5.12-2+apertis0 (v2025pre)
- 65 • Apertis: dash 0.5.12-2+apertis0~v2024 (v2024)

66 **Backport of Apertis update for Debian package**

67 In case of Apertis specific updates, there are different scenarios. The first one
68 and simpler is that the older release was matching the same version, in which
69 case the exact same version will be applied. The second one, is when there are
70 different Debian version. This is also simple, in the sense that the update needs

71 to be applied on top of the older version and the Apertis revision needs to be
72 increased.

73 **Example:**

74 Starting point:

- 75 • Apertis: dash 0.5.12-2+apertis0 (v2025pre)
- 76 • Apertis: dash 0.5.12-1+apertis0 (v2024)

77 After update:

- 78 • Apertis: dash 0.5.12-2+apertis1 (v2025pre)
- 79 • Apertis: dash 0.5.12-1+apertis1 (v2024)

80 The third case is when there version in the older release is the same Debian
81 version but an older Apertis revision. In this case we cannot increment the
82 Apertis suffix in the older release since that version already exists, so an addition
83 suffix needs to be added.

84 **Example:**

85 Starting point:

- 86 • Apertis: dash 0.5.12-2+apertis1 (v2025pre)
- 87 • Apertis: dash 0.5.12-2+apertis0 (v2024)

88 After update:

- 89 • Apertis: dash 0.5.12-2+apertis2 (v2025pre)
- 90 • Apertis: dash 0.5.12-1+apertis1+v2024 (v2024)

91 **Reference**

- 92 • [Debian policy for package versioning¹](https://www.debian.org/doc/debian-policy/ch-controlfields.html#version): it contains a comprehensive expla-
93 nation of how the versions work.
- 94 • [Debian Go policy for package versioning²](https://go-team.pages.debian.net/packaging.html#_version_numbers): it provides a guide for generat-
95 ing a version when upstream does not use version numbers (for instance,
96 it's common for Go libraries to have no version numbers in the traditional
97 sense).

¹<https://www.debian.org/doc/debian-policy/ch-controlfields.html#version>

²https://go-team.pages.debian.net/packaging.html#_version_numbers