ePlanning Program

ePlanning API Versioning Policy

Release 1.0

January 2020

Published by NSW Department of Planning, Industry and Environment

dpie.nsw.gov.au

Title: ePlanning API Versioning Policy

Subtitle: Release 1.0

First published: 13-Dec-19

1 Document change history

Date	Version	Change	Author(s)
21-Jan-2020	0.1	Draft version	Andrew Seymour
20-Apr-2020	1.0	Release version	Andrew Seymour

1.1 Related documents

Related document title	Author(s)

2 Table of Contents

Contents

3 Versioning Strategy

3.1 Overview

API versioning follows a 3-number version number format, with each number separated by a period.

The version number consists of MAJOR.MINOR.PATCH information where:

MAJOR version update which includes breaking API changes,

MINOR version when functionality is added in a backwards-compatible manner,

PATCH version is used for backwards-compatible bug fixes.

Example release cycle:

API	Version	Notes
DAMgmt	1.0.0	Initial release
DAMgmt	1.0.1	Bug fix
DAMgmt	1.0.2	Bug fix
DAMgmt	1.1.0	Feature enhancement
DAMgmt	2.0.0	New Major version

3.2 Version notation in APIs

Only the Major version number is present in the URL for the API to denote the API version as Minor and Patch versions are considered transparent to the consumer. In the example below, the v1 denotes the major version of v1 for the DA Management API.

https://api.apps1.nsw.gov.au/planning/DAMgmt/v1/AcceptReturn/PAN-XXXX

when version v2 is released the URL will change to

https://api.apps1.nsw.gov.au/planning/DAMgmt/v2/AcceptReturn/PAN-XXXX

4 Backwards Compatibility

4.1 Backwards compatible (non-breaking) changes

Non breaking changes are defined as changes to the API interface or functionality that do not have an adverse effect on the consuming application. These changes may be bug fixes or functional enhancements that do not impact the usability of the API for existing consumers. Consumers wishing to take advantage of new functionality in a new minor version will need to upgrade their system but they are not required to do so.

Examples of non-breaking changes include (but are not limited to):

- Adding an API interface to an API service
- Adding a method to an API interface
- Adding an HTTP binding to a method
- Adding a field to a request message
- Adding a field to a response message
- Adding a value to an enumeration (inbound API only)
- Adding an output-only resource field

4.2 Backwards incompatible (breaking) changes

Breaking changes are changes that will make the API unusable for existing consuming systems, requiring a change to the consuming system, including a change to the API endpoint.

Examples of breaking changes include (but are not limited to):

- Removing or renaming a service, interface, field, method or enumeration value
- Changing an HTTP binding
- Changing the data type of a field
- Changing a resource name format
- Changing visible behaviour of existing requests
- Changing the URL format in the HTTP definition
- Adding a read/write field to a resource message

5 API Lifecycle

5.1 Concurrent Versions

DPIE will maintain two concurrent major versions of each API. These are defined as version 'n' and 'n-1'. Each major version will have one, and only one, minor/patch version.

When new version is release, the current version (n) will be demoted to a deprecated version (n-1). For example, we have 2 versions v1 and v2. When v3 is release as the current version, v2 will be considered as deprecated and v1 will be decommissioned.

5.2 Minor versions

New minor versions may be released at any time in line with the Departments development timeline and release plans.

5.3 Deprecation and Decommissioning

When an API version is superseded and it becomes the 'n-1' version, it is considered deprecated. Existing consumers of this version will have a minimum of 12 months to upgrade to the newer version of the API. After 12 months, the API version will be decommissioned and can no longer be used.

New API consumers should not start using a deprecated version and should always start using latest version of an API.