Web development practices change over time, and jQuery intends to change as well so that it can serve as a better library for developing web sites. This has already been done several times in the past, such as the jQuery 1.9/2.0 release where many questionable APIs were removed. In general the team considers the following things when making changes:

- Does the change improve developer practices? The design of some APIs can inadvertently cause developers to use poor practices that reduce maintainability or harm performance. API changes ideally encourage good practices.
- What is the impact of change or removal? jQuery is used by nearly every web site, so just about any change will cause *something* to break on some web sites. However, the team does not want to cause undue hardship in upgrades unless it is balanced by benefits.
- Will this change cause exceptions that previously did not occur? jQuery's API guidelines say that invalid inputs yield undefined results. Although that is still an important guideline, there may be situations where existing code expects a no-op from such situations. These should be considered in the impact evaluation.
- Can the change be warned and filled in the jQuery Migrate plugin? It becomes much easier for developers to upgrade their code when Migrate can detect and warn about uses of changed, deprecated, or removed features.