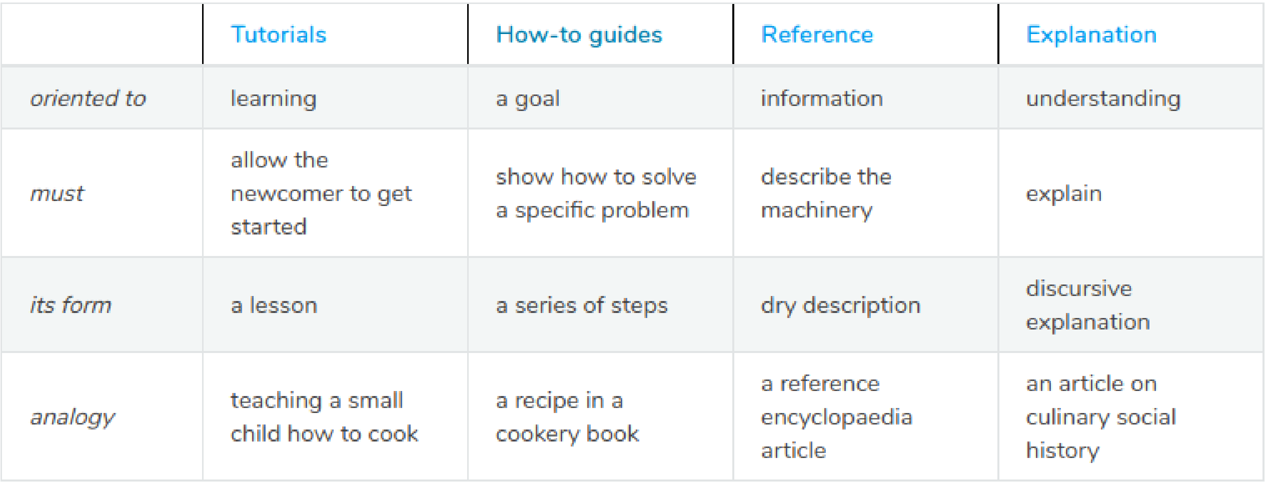
Notes on Technical Writing

In writing technical articles in the Knowledge Base I try to adhere to certain principles and patterns, which I will attempt to justify here.

When it comes to technical writing, most of my articles are how-to guides per this useful system of documentation classification summarized in this table:



	Tutorials	How-to guides	Reference	Explanation
oriented to	learning	a goal	information	understanding
must	allow the newcomer to get started	show how to solve a specific problem	describe the machinery	explain
its form	a lesson	a series of steps	dry description	discursive explanation
analogy	teaching a small child how to cook	a recipe in a cookery book	a reference encyclopaedia article	an article on culinary social history

How-To guides tell you how to solve a problem or achieve a goal. They are not tutorials, or deep dive explanations, or references.

Respect the Reader's Time

- How-To guides should have an introductory paragraph describing the problem and outline of the solution, allowing the reader to tell quickly if the article is relevant to them.
- No build-ups. Spill the beans quickly. Get to the good stuff as fast as reasonably possible, then "back up" in later paragraphs to fill in the details.
- Liberal use of bullet points, which are easier to scan than dense paragraphs.

  Sometimes a deep dive into a tangentially related topic is useful. Tangents are allowed, but should be hidden in an expanding macro, like this, or in a clearly distinct side panel.

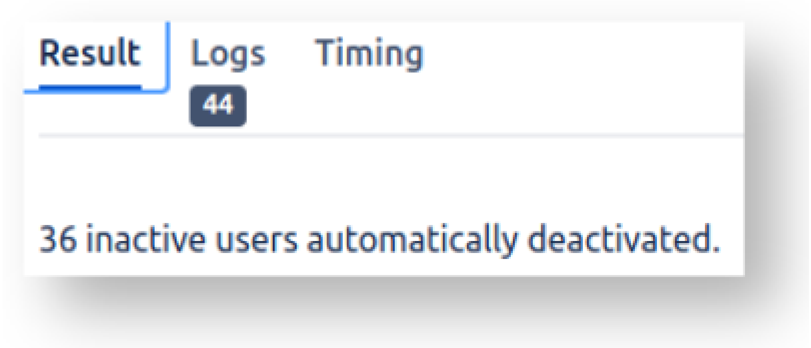
Assume an advanced readership

Articles are aimed at an "advanced" administrator, who knows in general how Jira works, what a plugin is, has a vague understanding of SQL and databases. How-to articles are goal-oriented, not learning oriented; thus, for example, an article might state "run the following SQL" without describing how you are to load up, say, `psql` and connect it to your Jira database.

Full Size Screenshots

A pet peeve of mine is tiny screenshots. If a screenshot is important, I want to see it properly! Yes, it looks slightly amateurish having a giant screenshots embedded in a page, but that's too bad; aesthetics will have to take a back seat to readability.

To distinguish large screenshots from the rest of the page, you will notice extensive use of the drop-shadow effect, e.g.:



36 inactive users automatically deactivated.

Again, slightly cheesy but function over form.

Terminology

Plugins, not 'apps' or 'add-ons'

Originally Jira had 'plugins'. Then they were renamed 'add-ons' for a few years, and now they're called 'apps'. To quote Stephen Deutsch's explanation:

> *Just to add to this, originally plugin was the term for a piece of software that extended the functionality of server products, as it is a piece of software written in Java that is installed inside of the host product (Jira, Confluence, etc.).*
>
> *When this functionality was brought to Confluence Cloud via Atlassian Connect, the software that extended the functionality existed on another server in any language, so the term plugin didn't make as much sense in that context. That's why they decided to use the term add-on.*
>
> *Finally, in order to unify the naming convention (as well as introduce terms that many consumers were already used to like app and appstore), they decided to change the term for any piece of software that extended functionality to be called an app. This change was made around August of 2017: https://community.developer.atlassian.com/t/marketplace-now-has-apps/7524*
>
> *Whether that's a good thing or not is a separate discussion...*

Articles on this site I work mostly with Jira Server, where 'plugin' is the original term. Unlike 'app', 'plugin' correctly denotes the embedded, subordinate, extending nature of the extension. Thus, I will stick to 'plugin' while Server is my preferred platform.