

Technical Writing Style Guide



- Defines standards for clear, consistent software documentation



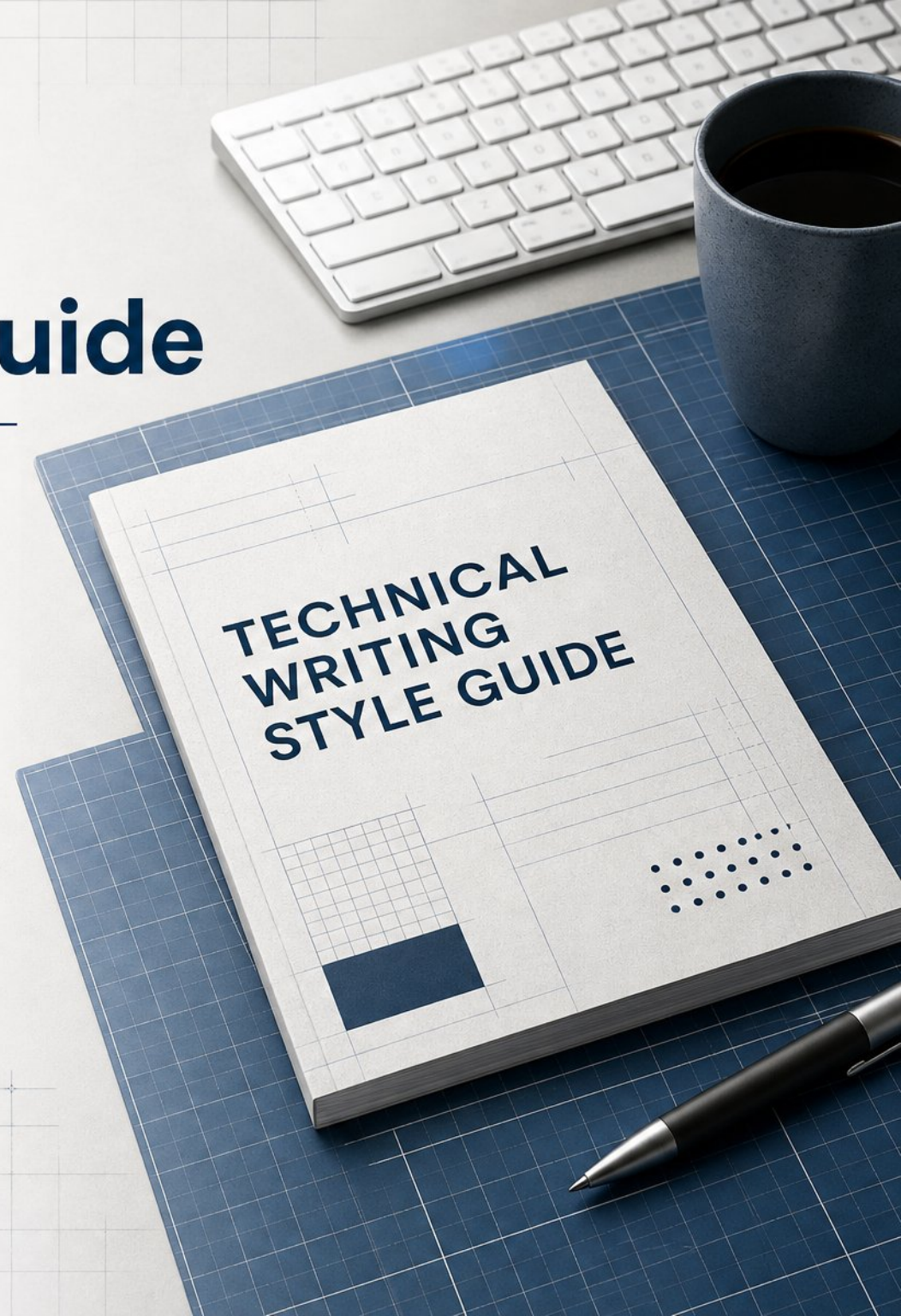
- Covers voice, terminology, formatting, UI text, and code samples



- Aligns writers, developers, and product documentation leads



- Roadmap: write, review, and enforce documentation consistency



Core Principles of Effective Technical Writing



- **Clarity:** one idea per sentence, direct language



- **Conciseness:** short, focused sentences without filler



- **Consistency:** align terms, formatting, and structure

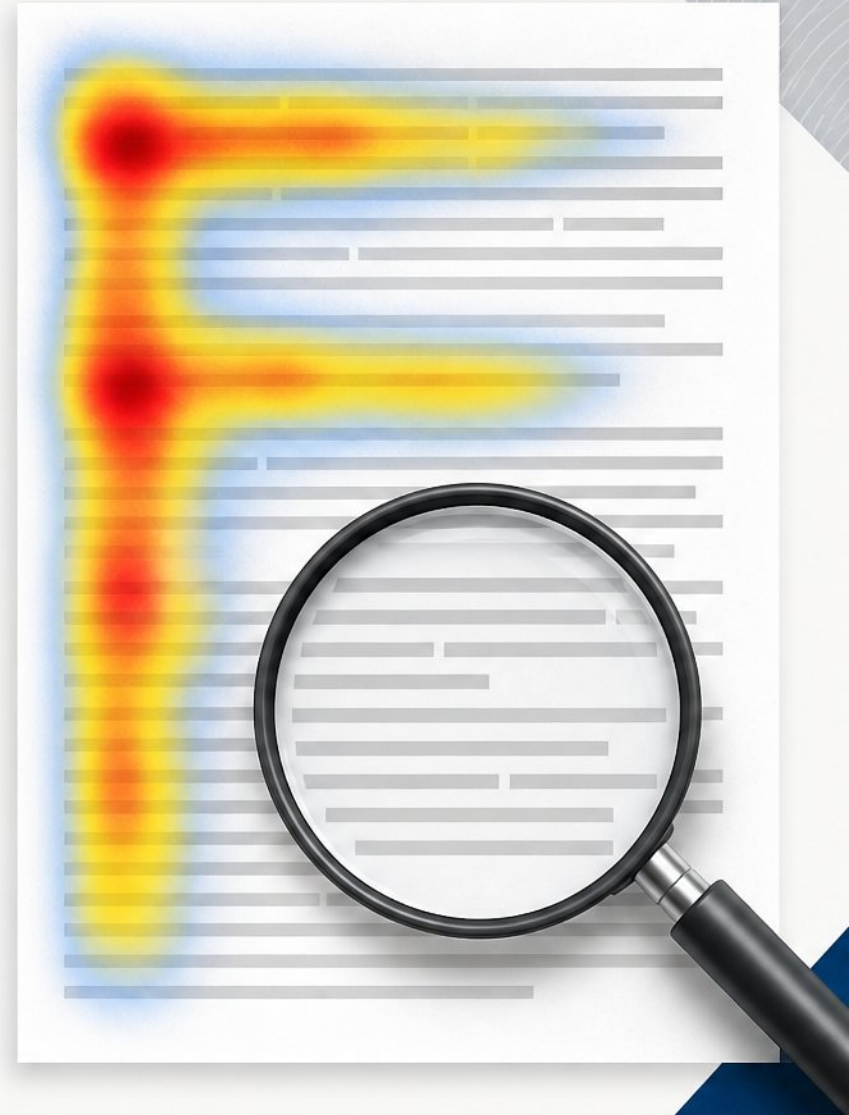


- **Audience-first:** match content to reader expertise



Readability and the Global Audience

- Aim for 15–20 words per sentence, one idea per sentence
- Front-load key information for F-shaped scanning patterns
- Use plain language, avoid idioms and cultural references
- Balance precision with accessibility for localization



Voice and Tone

- Keep one consistent voice, adapt tone to context
- Default: active voice, second person, present tense
- Tutorials warm; references neutral; troubleshooting direct
- Avoid buzzwords, excessive politeness, stiff or casual prose

BEFORE

Stiff & Formal



Please be advised that this guide contains the necessary information to assist with the completion of the task.

You are required to review the instructions in their entirety before proceeding.

In the event of an issue, please consult the related documentation.

Ensure that all steps are followed in the order presented.



AFTER

Warm & Helpful



This guide shows you how to complete the task.

Read all the instructions before you begin.

If you run into an issue, check the related documentation.

Follow the steps in the order shown.



Grammar and Mechanics



- Consistent capitalization, punctuation, and spelling



- Use active voice and imperative mood in procedures



- Avoid common mistakes like dangling modifiers



- Define abbreviations, use unambiguous dates



Terminology Management



- Maintain an approved glossary as the single source of truth



- Avoid jargon, ambiguous terms, and unnecessary synonyms



- Track deprecated terms and platform-specific names explicitly



- Align decisions across writers, developers, and translators



- Review and update terminology continuously



APPROVED TERM

Authenticate



DEFINITION

To verify the identity of a user, system, or process before granting access to a resource.



REJECTED SYNONYMS

- × Log in
- × Sign on
- × Verify user

Document Structure and Formatting



- Use descriptive headings that match the content beneath



- Keep a clear hierarchy: H1 → H2 → H3 without skipping levels



- Choose lists or tables based on the information type; add a lead-in sentence



- Emphasize with **bold** (UI), *italics* (terms), and `code` font consistently

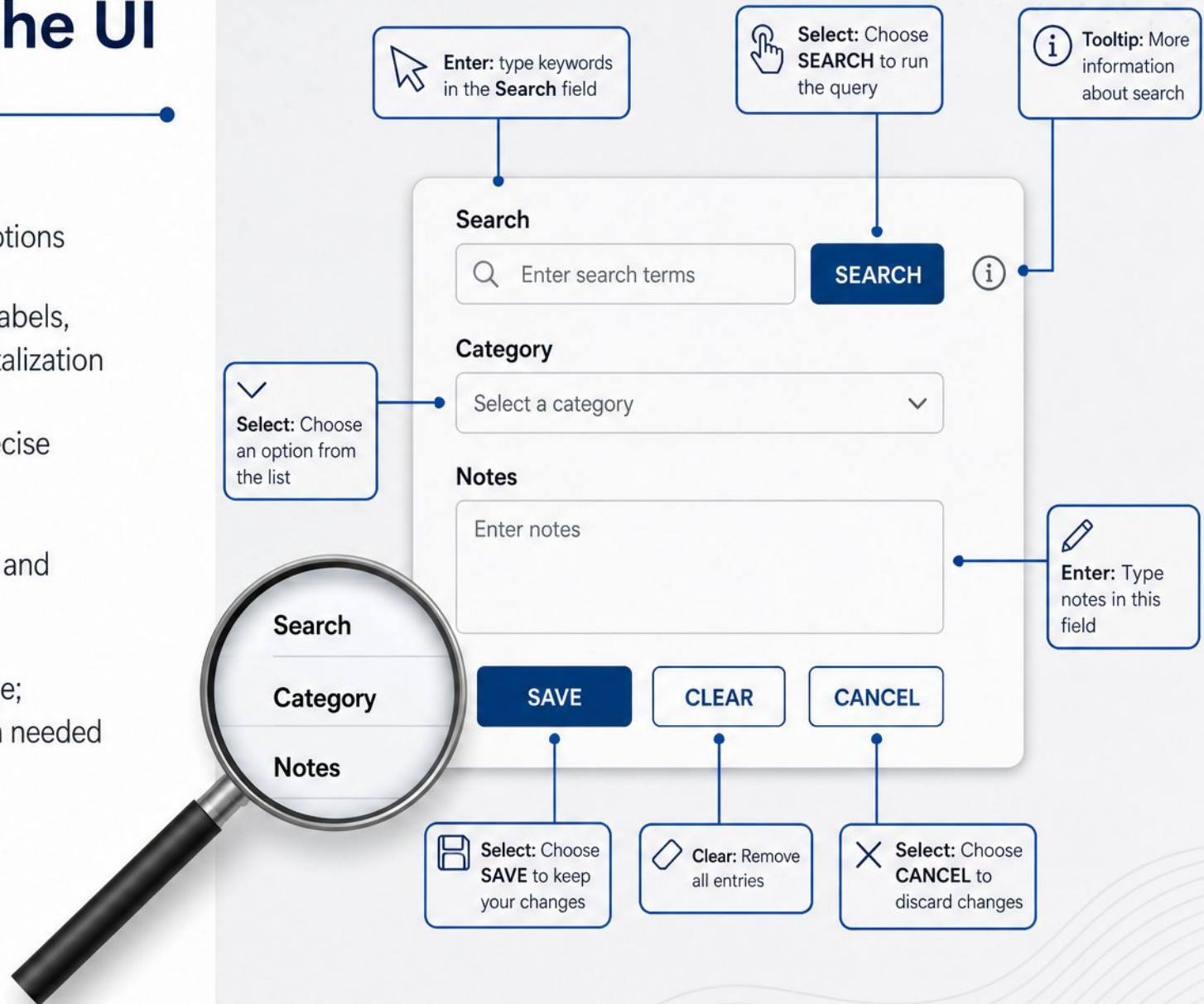


- Reuse templates for common formats to maintain consistency across the corpus



Writing for the UI

- Focus on user goals, not widget-by-widget descriptions
- Use bold for UI element labels, matching on-screen capitalization
- Describe actions with precise verbs: select, enter, clear
- Include accessibility cues and tooltips for icons
- Avoid directional language; provide screenshots when needed



Code Samples and API References



- Consistent formatting for inline code, placeholders, and commands



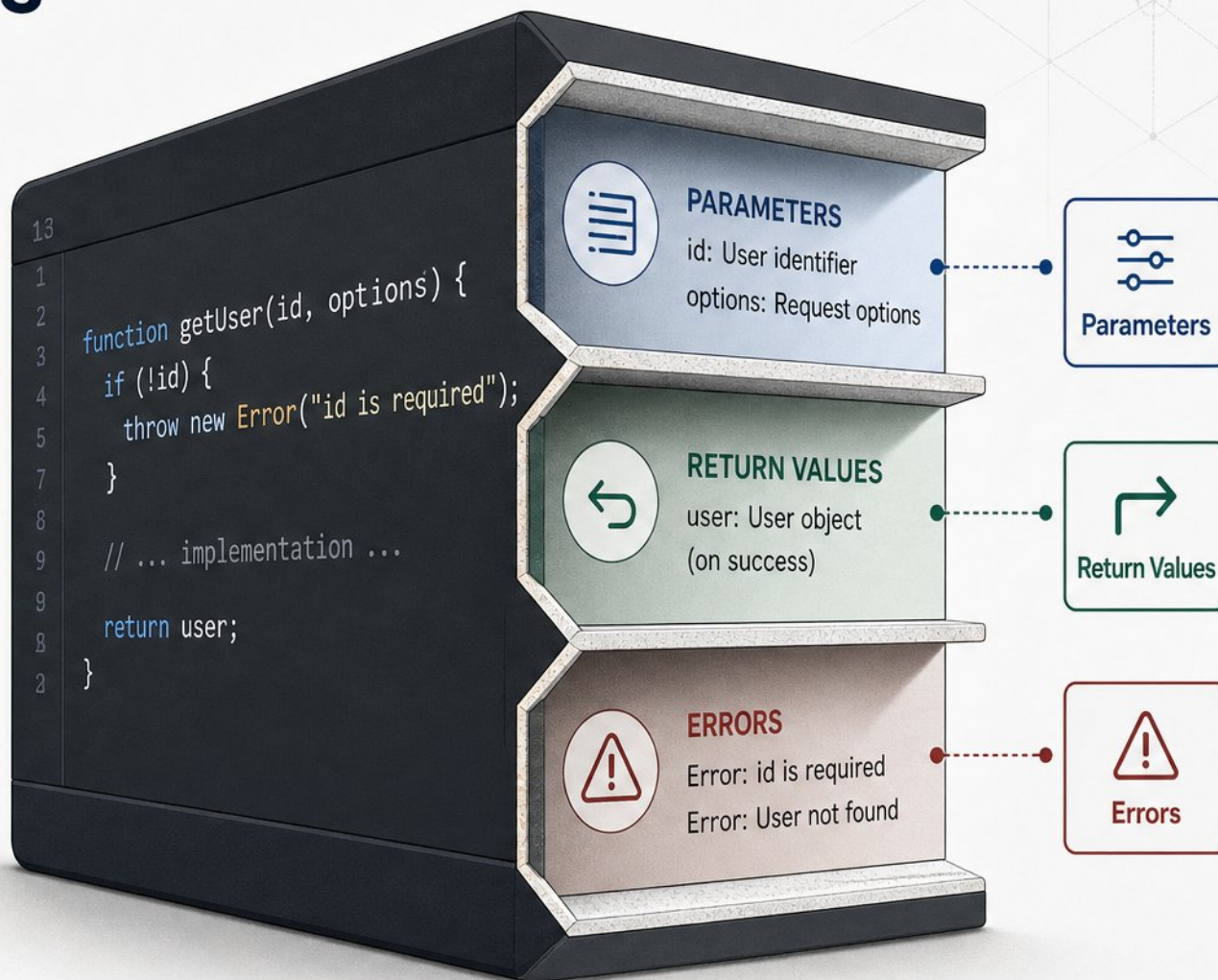
- Complete, runnable examples with expected outputs



- Clear explanations of parameters, return values, and errors



- Accurate cross-references between prose and code



Accessibility in Documentation



Descriptive Link Text

Learn more about [accessible headings](#).



Alt Text

Alt text: Person using a keyboard with a screen reader.



Proper Heading Structure

Clear hierarchy helps everyone navigate content.



Inclusive Language

Use people-first, respectful, and inclusive terms.



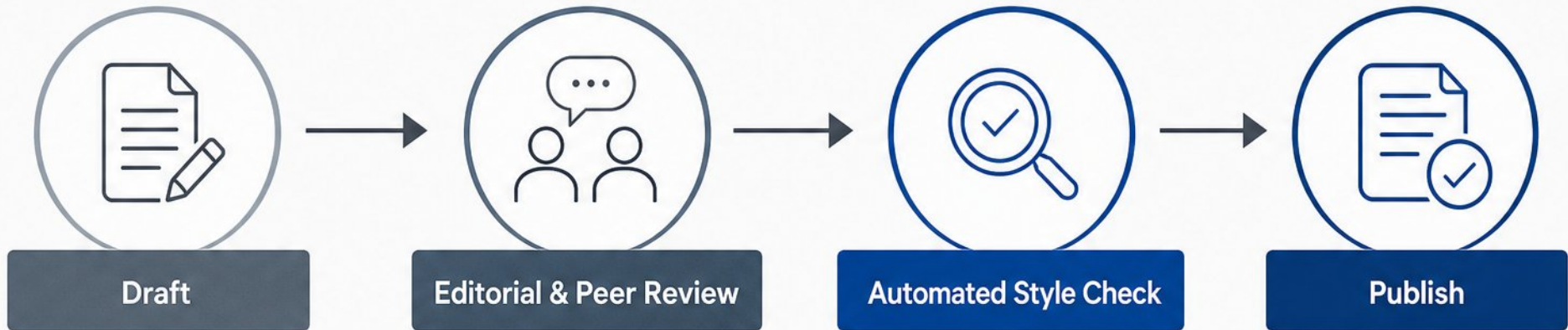
Assistive Technology

Ensure compatibility with keyboard and screen readers.

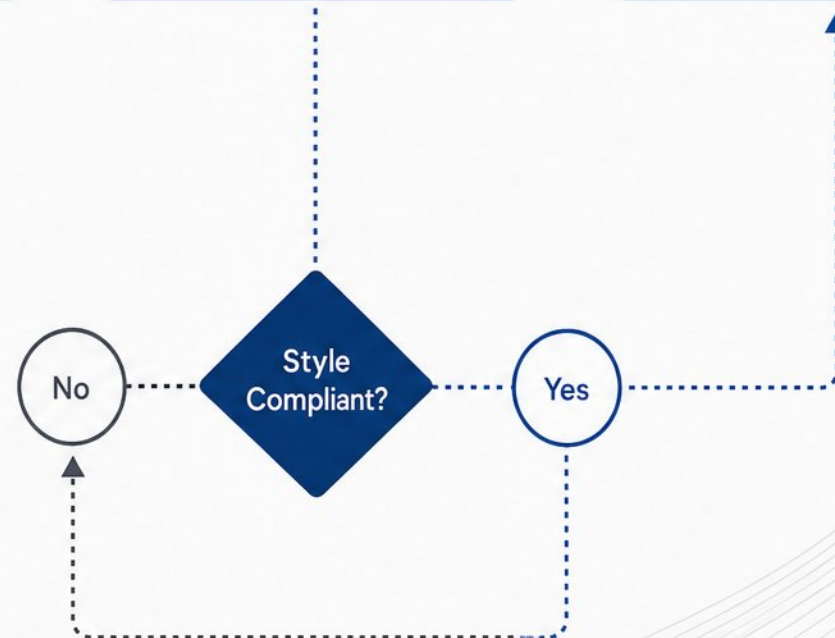


- Use descriptive headings and meaningful link text
- Write alt text for all images
- Avoid directional and visual-only cues
- Use inclusive language, avoid ableist terms
- Ensure keyboard and screen reader compatibility

Applying the Style Guide in Workflows



- Integrate style checks into editorial and peer review
- Use linters like Vale to catch surface issues automatically
- Collaborate closely between writers and developers
- Handle exceptions and local deviations consistently



Automated Style Checking

- Vale lints prose like ESLint lints code
- Rules target sentences, headings, and lists
- Configure vocabularies and severity levels in CI/CD
- Pair rule-based tools with LLM review for context
- Start with warnings, tighten enforcement as noise drops



Industry Style Guides as References



- Google, Microsoft, Apple, and GitLab offer public guides



- Consensus exists: plain language, active voice, second person



- Google and Microsoft are the most popular starting points



- Adapt a base guide; avoid over-standardizing



- Use third-party references like Merriam-Webster for spelling

Practical Adoption and Continuous Improvement



- Build a living style guide that evolves with your team



- Drive adoption through examples, templates, and automated checks



- Measure consistency and reduce style debt over time



- Balance strictness with readability and domain judgment



- Start with a base guide like Google or Microsoft, then customize



PersonWise.

PersonWise • AI digital-human course platform



Scan the code or click anywhere to watch this course live, with voice Q&A

personwise.ai/t/68cc812c/public