Technical Writer interview questions

careertips.com.au: common questions and what they're really testing

What to expect

Technical Writer interviews usually mix questions about your writing and research process with questions that test how you work with engineers and other subject matter experts under real deadlines. Expect a practical focus: how you gather information, structure it, and keep it accurate as products change.

- **Process:** Questions asking you to walk through how you actually produce a piece of documentation, from research to publication.
- **Behavioural:** Questions about past situations, particularly working with difficult or time-poor subject matter experts and stakeholders.
- **Technical:** Questions checking hands-on familiarity with tools like Git, Confluence, JIRA, MadCap Flare or FrameMaker, and with docs-as-code workflows.
- **Scenario:** Hypothetical situations testing judgement when requirements change late or information is incomplete.
- **Editorial/quality:** Questions about how you maintain accuracy, consistency and audience-appropriate tone across a documentation set.

Most interviews open with questions about your background and the types of documentation you've produced, move into process and technical questions to test practical capability, then use one or two scenario or behavioural questions to see how you handle ambiguity or stakeholder friction. Some employers ask for a writing sample or a short editing exercise either before or after the interview.

1. Walk me through your process for creating a new piece of API documentation from scratch.

Why they ask

This checks whether you have a repeatable, professional process rather than an ad hoc approach, and whether you understand what developers actually need from API docs.

How to structure your answer

Answer as a sequential walkthrough: describe each stage in order (scoping the feature, interviewing the developer, drafting, testing code samples, review, publishing) and note what you check for at each step.

Example answer

“I start by scoping what the endpoint or feature actually does, usually from a ticket in JIRA and a look at the existing codebase if I have access. Then I set up a short interview with the developer to confirm request and response formats, authentication, and any edge cases they know will trip people up. I draft the reference content and write a working code sample myself, testing it against a live or staging environment rather than just trusting what I was told. Once it reads clearly and the sample runs, I send it for technical review to the developer, incorporate their corrections, and publish it through our version-controlled docs repository so it ships alongside the release.”

2. Tell me about a time you had to get technical information out of a subject matter expert who was too busy or reluctant to help.

Why they ask

Getting accurate information from developers and engineers who don't see documentation as a priority is one of the most common friction points in this job.

How to structure your answer

Use STAR: describe the situation, the specific task you needed from them, the action you took to get their time and cooperation, and the result.

Example answer

“I once needed sign-off on a set of procedural steps from a senior engineer who was heads-down on a release and kept pushing my emails aside. Rather than keep chasing by email, I booked a fifteen-minute slot directly in his calendar, framed it as a quick read-through rather than a meeting, and had specific yes/no questions ready so he could get through it fast. He reviewed the steps in the slot, flagged two inaccuracies I hadn't caught, and I published the corrected guide that week instead of it slipping past the release.”

3. How do you use Git or version control in your documentation workflow?

Why they ask

Many technical writing roles now use docs-as-code workflows, so employers want to know you can work alongside developers in their existing tooling rather than needing a separate system.

How to structure your answer

Explain the concept in your own words first, then give a concrete example of how you've applied it.

Example answer

“I treat documentation source files the same way developers treat code: I work in branches, commit changes with clear messages, and raise a pull request for review before merging into the main documentation branch. In practice, that meant when I updated a set of guides for a new release, I could work in a feature branch alongside the developers' own feature branch, and our docs went live through the same review and merge process as their code, so nothing published out of sync with the actual release.”

4. A product feature changes right before release and your documentation is already written. What do you do?

Why they ask

This tests judgement under time pressure and whether you prioritise accuracy over just meeting a deadline.

How to structure your answer

Answer as a judgement call: state the immediate priority, the trade-off you're weighing, and the decision you'd make, with reasoning.

Example answer

“My first priority is finding out exactly what changed and how it affects what I've already written, so I'd go straight to the developer rather than guess. If the change is small, I'd update the affected sections and flag to the release manager that docs need a final pass before go-live. If the change is significant and I can't verify it in time, I'd rather publish with a clearly marked 'pending update' note on that section than ship inaccurate steps, because a wrong instruction causes more support load than an honest gap.”

5. How do you decide the right structure and level of detail for different audiences, like end users versus system administrators?

Why they ask

This checks whether you think about audience needs rather than writing one version of a document for everyone.

How to structure your answer

Explain your reasoning process, then illustrate with a specific example of adapting content for two different audiences.

Example answer

“I start by asking what the reader is trying to accomplish and how much context they already have. An end user guide needs plain language, screenshots and minimal jargon, focused on completing a task. A system administrator guide can assume technical background and needs more detail on configuration options, permissions and failure scenarios. For one product I maintained a shared source of core information but split it into a task-based user guide and a more technical admin reference, so each audience wasn't wading through content that wasn't relevant to them.”

6. How do you handle a disagreement with a developer about whether the documented behaviour is actually correct?

Why they ask

Writers often catch inconsistencies between documented and actual product behaviour, and employers want to know you'll push for accuracy without damaging the working relationship.

How to structure your answer

Answer as a behavioural example, focusing on how you verified the issue and resolved the disagreement professionally.

Example answer

“I once documented a setting based on the specification, but a developer told me the behaviour had changed and the spec was outdated. Rather than take either version at face value, I tested the setting myself in a staging environment and showed the developer the actual result. It turned out the behaviour had partially changed, so we agreed on the correct wording together and I updated the spec reference as well as the docs, so the next person wouldn't hit the same mismatch.”