Personas in Writing

By Matt Eland on January 30, 2020 in Communication

Some software engineering organizations rely on something called User Personas to help the organization keep users in perspective. It's a decent concept, but I think it can be applied to more areas.

In this article, I'll explore the idea of using personas as a writer to keep your readers in mind as you select articles and structure your thoughts.

Note: If you follow me for software development content in general, this piece may not be for you, though I do explore why I write and who my target audience and what my primary goals are near the end for those curious.

What is a Persona?

Photos by Armin Lotfi, Mohammed Hassan and Irene Strong on Unsplash

A persona is an easy to understand simple description representing a subset of a larger target audience.

Organizations often display these personas prominently on boards near team areas, in software tools, on whiteboards, or in other areas. The intent is to keep personas at the edge of every decision we make, in order to make sure we don't forget users while solving technical problems or making product decisions.

A persona should be brief, clear, and quickly communicate the behavior and needs of the user to your development team. Examples for a medical system might include:

- A busy single mom who needs to quickly find healthcare information
- A college student on a budget trying to find the best possible rates
- A caregiver for someone with a chronic medical condition, looking for the best possible care

Using these personas, the team can quickly evaluate every decision they make – particularly those affecting the user interface. This helps the team optimize the system for their users.

Personas in Writing

You can take the concept of personas and apply it to writing. Instead of designing a system around personas, you're selecting articles based on them, then tailoring your approach to the topic to meet their needs.

Some sample audience personas in technical writing might include:

- A skilled professional looking to keep their knowledge up to date with emerging trends
- An expert looking for insight deep into the internals of a specific language or library
- A programmer transitioning into a software architect role

So, how many do you pick? I think that's up to you based on how focused and specific you'd like your audience to be. If you're writing a single article, you have a lot more leeway to gear it towards a single persona than someone would if they were writing a larger series or trying to build a web site of related content.

Generally, I'd encourage you to pick no more than 5 personas for a generalist type of approach and no more than 2 or 3 personas for a very focused audience.

I should note as well that not all personas need to get equal attention. It's okay for one to be your core audience while others are peripheral that you keep in mind while writing. The point is to identify who you care about.

Why are you Writing at all?

Before you pick your target personas, you need to back up a moment and ask yourself this high level question:

Why are you writing at all?

For me, a part of the answer to this question became clear to me this past week:

I'm writing because I personally feel that software development has a high barrier to entry and this barrier is causing problems.

I'm not talking about the basics of "this is a for loop and these are variables".

I'm talking about things like "How do I avoid having to change every file in a program when making a simple change" or "Here's how to make it impossible for bugs to thrive in your systems" or even "This is what functional programming has to offer".

In software development, we have a habit of performing *litmus tests* that effectively say "You're not a real developer unless…":

- You know the difference between `const` / `let` / `var` or what is truthy and falsy in JavaScript
- You can describe Big O notation in under a minute
- You don't get frightened by terms like dependency injection or cyclomatic complexity
- You've been known to say "adapter builder factory" in your sleep
- You know the deep internal details of how threading or garbage collection works in your language of choice
- You've made at least one junior developer cry during a code review

Folks, these types of measurements are dangerous.

People struggle with impostor syndrome every day, and let's face it: the software development profession has a very severe diversity problem; Just over 6% of my readership is female and that's actually *high* in the software development world.

Yes, it's good to know the ins and outs of a language. Yes it's good to know standard industry terms. But you have to realize that people are entering the field in different ways now via boot camps and other avenues than they were a decade ago.

Yes, this new breed of coder may not have had as much theory or practice but they work hard, are passionate, and want to make a difference. Let's not shame them for it, but instead build up their skills, knowledge, and comfort to help them succeed and contribute more.

And that's a part of what I'm all about.

Meet my Target Personas

Let me tell you who I'm writing for by introducing you to my personas:

Betsy the Bootcamp Graduate

Photo by NESA by Makers on Unsplash

Betsy is someone new to the development field. She understands the basics of programming, but the dynamics of development organizations are new to her. She's starting to understand how broad and deep the field truly is as well as how *impossible* it is to know everything.

For Betsy, I want to stress software engineering fundamentals in simple ways. I want to equip her to make changes with the confidence that she's not breaking anything. I want to help her understand the senior people on her team and the way they talk and think. I want to introduce her to new concepts and get her comfortable with things with frightening names (I'm looking at you, cyclomatic complexity).

Betsy benefits greatly from simple tutorials of getting started with new technologies or libraries as well as conceptual overviews.

Sara the Not-Quite Senior Developer

Photo by Mimi Thian on Unsplash

This is someone a little further along in their career. They understand how to do software development tasks and can accomplish tasks assigned to them without regularly breaking things, but are learning to think about bigger picture things such as how code changes impact maintainability and testability. They want to build up the knowledge, perspective, and comfort they need to reach the next level.

For Sara, I focus on comparing and contrasting options, talking about tradeoffs of various decisions and technologies, talking about code maintainability, and some of the more conceptual aspects of software engineering teams.

Simple, low intimidation discussions of software architecture and new features in languages are also helpful for this persona as well as deep dives into specific areas they're already generally familiar with.

Priya the Prodigy

Photo by Mimi Thian on Unsplash

Priya is someone who loves development and has a knack for it. She’s doing well, enjoying it, and learning on the side. She’s excelling and wants to excel and grow, but doesn’t always know what’s out there to explore.

For the Priyas out there, the best thing to do is introduce them to options to investigate and play with at their leisure as well as some general guidance such as “you might want to consider this if…” or “doing this can have these tradeoffs…”.

For Priya, I focus on exploring radical new concepts or combinations of technologies (genetic algorithms applied to squirrel brains comes to mind).

Priya is also more sensitive to industry trends such as exploring functional programming, document databases, or microservice architectures.

In general, Priya is an explorer who wants to join an expedition. My role is to give her encouragement, a map, and a shovel and tell her where the power lines are buried, then tell her to have fun and let me know what she finds.

Matt the Manager

Me, managing
Strange, this guy has the same name and job as I do.

Here I’m writing to the past version of me. I’m writing to Matt the senior developer who was considering management. I’m writing to Matt the new manager and trying to stress things that have helped me succeed while offering cautions on ways I’ve nearly failed.

And I’m also writing to those out there in management and general business roles who work with developers who want to understand more about who they are, what they care about, and the nature of the various beasts they have to wrestle to ship quality software on time.

For this persona I focus on teams, projects, trends over time, and the macro scale of things. I also try to keep things at a very readable level instead of talking about specific technical terms for the non-technical managers out there.

Tina the Tester

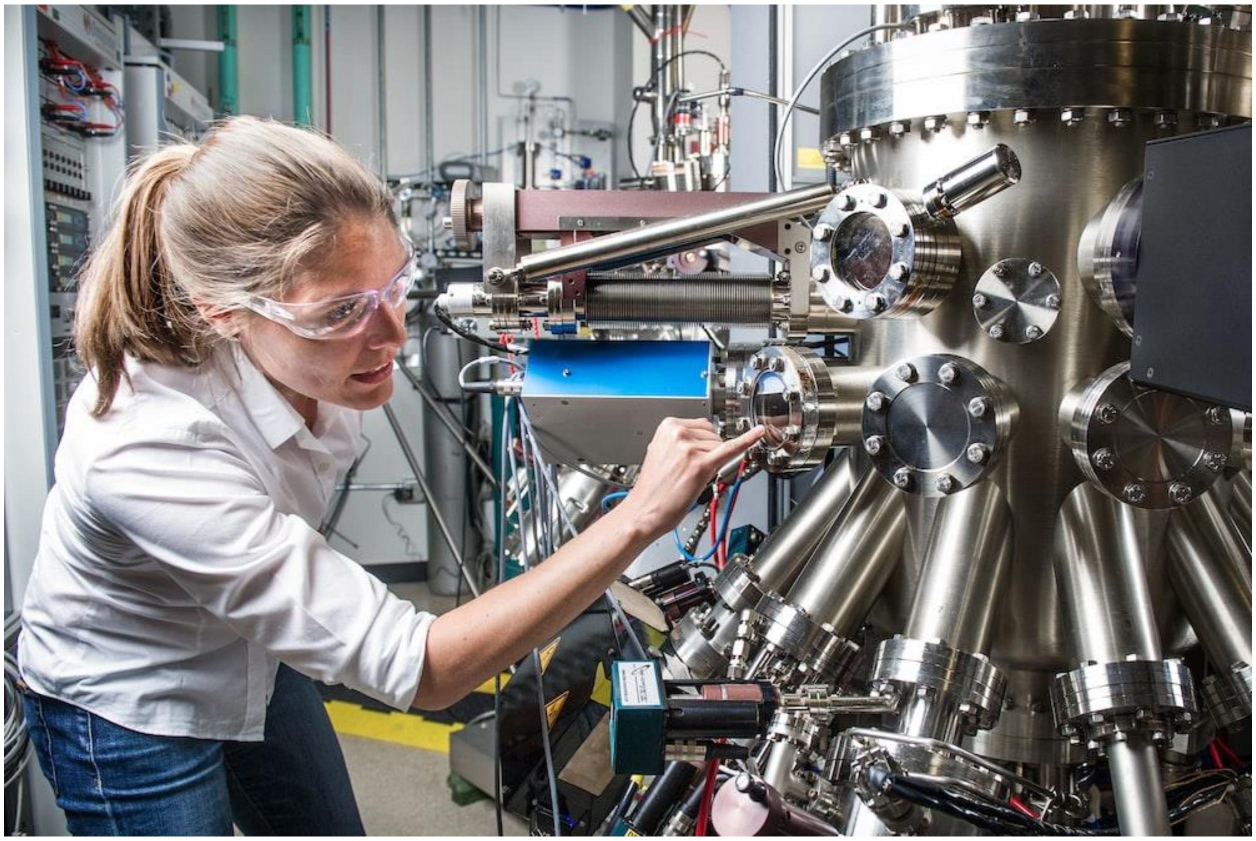
Photo by Science in HD on Unsplash

I write on software quality. This means that testers interested in ensuring that quality at the quality assurance level are bound to come into my readership, even though I primarily gear myself towards developers.

Like some of the other groups, testers benefit from me keeping things at a very simple level in terms of descriptions, etc.

Unlike some of the other groups, testers benefit from exploring the various ways that things can fail – particularly with specific fictional examples.

Testers aren’t my primary audience, but I want to keep them in mind when exploring concepts related to software quality for obvious reasons.

Applying Personas

So, now that we’ve talked about personas in software engineering and in writing, what does it look like to work with personas as a writer?

You can use personas when selecting what to write about. For example, when evaluating ideas for articles, I can look at my list of personas and say “How much would X be interested in reading this?” or “what would Y have to gain from this article?”

Additionally, you can use each persona as a lens of sorts when reviewing drafts of your content. Something that might make excellent sense to Priya may be confusing to Betsy and need additional explanations or visuals, for example.

And sometimes, it’s okay if you write off a specific persona or two when creating an article. You can’t be all things to all people, but keep in mind the people you’re trying to serve and *why* you care.

Closing Thoughts

I should put a disclaimer here at this point. I have not followed my own advice. My 5 personas are unlikely to be interested in this specific article, let alone the other 90 I’ve written these past 4 months.

That’s because I wrote this article for myself. I wanted to examine and affirm my mission in writing to the community to a whole, and here it is:

My goal is to help aspiring and existing software professionals discover and understand new tools, processes, and ideas for creating quality software so that they can perform new tasks with confidence and excellence.

What’s your goal? Who is your target audience? Why do you care about them and what do you want them to learn?