what interests me keeps me engaged.

Dan: As Karthik said, there are a set of broadly-applicable principles and skills. I'd also argue that the way to really learn a technology is to build something with it that someone can actually use and then maintain it over time. I try to focus on technologies I actually need to use today, to solve a real problem. When you're learning something new for the sake of learning, make sure it's actually something new: One MVC framework will likely be similar to the next one, just as relational databases tend to be similar to each other, and so on. But a functional programming language might be fairly different from an object-oriented one, and a statically typed language will be different from a dynamically typed one. You'll learn more deeply by picking something truly different than what you're used to.

Klint: Conversely, how do you decide NOT to learn something? What makes you scratch something off the list or move it to the "not right now" category?

Dan: I usually only reach for a new tool if there's an overwhelmingly compelling reason to use it instead of what we're already using. So if there's no clear reason that the new thing is preferable, I say skip it and make your existing tools do the job. I'm a big advocate for having a small set of tools that you know well.

Monica: I agree. If something doesn't solve the sorts of problems I'm working on, I'm not going to spend time learning it. It's fun to learn for the sake of learning, but if I try to learn too many things at once I don't get the depth I want. To narrow things down, I think about how the things I want to learn complement what I'm working on and my goals. For example, I recently decided not to spend much time on emerging serverless databases. I've done some exploration of different options, but they don't really fit into the work I'm doing. For my personal projects, I'm able to leverage serverless functions instead of a database. It's not that I don't want to learn more about serverless databases eventually, but it's just not something I need immediately.

Karthik: I try not to write things off without learning a little about them first. I know that machine learning is a field that is advancing rapidly. Still, when I started with the beginner resources in ML, I felt that, because of the mathematics involved, it wasn't for me, so I ended up focusing on the data visualization side. But I gave ML a fair try first, and it's something I might come back to. Just because others are doing it doesn't mean that you have to do it if it doesn't grab your interest. Although I try not to be dismissive, it doesn't necessarily make a lot of sense to spend time on legacy technologies if you're not actively using them at work. It's worth paying attention to what sorts of platforms are being phased out.

When you're learning something new for the sake of learning, make sure it's actually something new. One MVC framework will likely be similar to the next one, just as relational databases tend to be similar to each other, and so on. But a functional programming language might be fairly different from an object-oriented one, and a statically typed language will be different from a dynamically typed one.

- Dan Kuebrich, VP of platform engineering at FullStory

Klint: What skills have you learned that ended up being the most transferable? What skills make learning other skills or technologies easier?

make learning other skills or technologies easier?

Karthik: The basics of computer science, like data structures and algorithms. I