**Getting to the Heart of Digital Accessibility**

by [Carie Fisher](https://alistapart.com/author/carie-fisher/)August 01, 2019



Illustration by [Dougal MacPherson](https://alistapart.com/author/dmacpherson/)

Quick! Think of the word “developer” or “coder” — what’s the first thing that comes to mind? Maybe a whiteish male in his twenties living in a busy metropolis, wearing a nerdy t-shirt and hoodie? Someone a bit like Mark Zuckerberg? Or maybe a younger Bill Gates or Sergey Brin? Any of the dudes from the HBO series *Silicon Valley*, perhaps? Certainly no one like me.

By tech standards, I’m old. I’m also female *and* a mother. I live in a midwestern town you’ve never heard of and will never visit — a town where the cows vastly outnumber the people. My hair color is (almost) natural and is no longer part of the ROYGBIV collection, so I have no perceived conference street cred. I own about a thousand geeky T-shirts, but never actually wear them in public, opting for more “girly” attire (or so was pointed out by a male colleague). On the surface, I look more suited to taking notes at a PTA meeting than writing code. I’m a bit of an outsider. A tech misfit.

So when my 11-year-old daughter finished her recent coding camp and excitedly declared, “Now I’m a real developer, Mom, just like you!” there was the usual parent pride, but also a small piece of me that cringed. Because, as much as I support the STEM fields, and want the next generation of girls to be coding wizard-unicorn-ninjas, I really don’t want my own daughter to be a developer. The rationale behind this bold (and maybe controversial) statement comes from a place of protection. The tech world we live in today is far from perfect. I’ve endured my share of misogyny, self-doubt, and sexual harassment. Why wouldn’t I want to protect her from all of that?

**The (diversity) elephant in the (computer) room**

You’ve heard this story before: there is not enough diversity in tech. This puzzling trend seems to continue year after year, even though numerous studies show that by including more people from underrepresented communities, a company can increase its [innovation](https://www.fastcompany.com/40515712/want-a-more-innovative-company-simple-hire-a-more-diverse-workforce), [employee retention](https://devskiller.com/diversity-in-tech/), and [bottom line](https://www.forbes.com/sites/annapowers/2018/06/27/a-study-finds-that-diverse-companies-produce-19-more-revenue/). Even with the recent push and supposed support for diversity and inclusivity from many Fortune 500 companies, women and female-identifying people still only hold [20% of all top tech jobs](https://smallbiztrends.com/2018/03/women-in-technology-statistics.html).

The data from FY 2018 shows that the number of women in technical roles at three of the top tech giants was [24% for Adobe](https://www.adobe.com/diversity/data.html), [26% for Google](https://diversity.google/annual-report/), and [22% for Facebook](https://newsroom.fb.com/news/2018/07/diversity-report/). While these numbers show that there is still not enough representation for women, these numbers do reflect a slight increase from the previous year (FY 2017: Adobe 22%, Google 25%, Facebook 15%). But even with this upward trend of hiring women in tech roles, the marginal growth rate has not caught up with the real world. The tech workforce is seriously out of touch with reality if, in 2019, a demographic (women) that represents more than half the global population is still considered a minority.

Sometimes this lack of diversity at the top level is blamed on a “pipeline” issue. The logic being: “If there are not enough girls who learn to code, then there will not be enough women who can code.” However, programs aimed at [teaching girls how to code](https://girlswhocode.com/) have skyrocketed in the past few years. Girls now make up about [half of the enrollment](https://csedu.gallup.com/home.aspx) in high-school coding classes and are scoring almost identically to their male classmates on standardized math and science tests, yet, young women make up only [18% of all Computer Science degrees](https://ngcproject.org/statistics). I have to wonder if this steep drop in interest has more to do with lack of representation in the tech sphere, than with girls and young women simply not being “smart enough” or “not interested” in working with code? At the very least, the lack of representation certainly doesn’t help.

Of course, the diversity picture becomes even more abysmal when you consider other underrepresented groups such as people of color, people from the LGBTQ community, and people with disabilities. And while I really don’t like glossing over these deeper diversity issues in tech, because they are abundant and are much more grotesque failings than the male/female ratio, I also don’t feel qualified to speak about these issues. I encourage you to look to and value the voices of others who can speak with higher authority on these deeper diversity issues, such as [Ire Aderinokun](https://twitter.com/ireaderinokun), [Taelur Alexis](https://twitter.com/TaelurAlexis), [Imani Barbarin](https://twitter.com/Imani_Barbarin), [Angie Jones](https://twitter.com/techgirl1908), [Fatima Khalid](https://twitter.com/sugaroverflow), [Tatiana Mac](https://twitter.com/TatianaTMac), [Charlie Owen](https://twitter.com/sonniesedge), [Cherry Rae](https://twitter.com/cherryrae), and so many others. And for those readers who are new to the topic of diversity in tech, watch Tatiana Mac’s recent conference talk [How Privilege Defines Performance](https://youtu.be/nQq_gZiZ-jg) — it’s well worth the 35 minutes of your life.

**The four stages in the digital accessibility journey**

However you look at it, the numbers don’t lie. There are some pretty significant diversity issues in tech. So how do we fix this issue before the next wave of young developers join the tech workforce? Simple: teach developers to write accessible code.

This may seem like a joke to some and stretch to others, but hear me out. When we talk about accessible code, what we are really talking about at its core is inclusiveness. The actual process of writing accessible code involves rules and standards, tests and tools; but inclusive development is more abstract than that. It’s a shift in thinking. And when we rethink our approach to development, we go beyond just the base level of simple code functionality. We instead think, *how* is this code consumed? How can we make it even more intelligible and easier for people to use? Inclusive development means making something valuable, not just accessible, to as many people as we can.

That line of thinking is a bit abstract, so let’s go through an example. Let’s say you are tasked with updating the color contrast between the text on a webpage or app and the background. What happens at each stage in the accessibility journey?

**Stage 1: Awareness**— You are brand new to digital accessibility and are still trying to understand what it is and how you can implement changes in your daily workflow. You may be aware that there is a set of [digital accessibility guidelines](https://www.w3.org/WAI/WCAG21/quickref/)that other developers follow, but you are a bit hazy on what it all means in a practical sense.

**Stage 2: Knowledge**— You know a bit more about digital accessibility and feel comfortable using a few testing tools, so you run an automated accessibility test on your website and it flags a possible issue with the color contrast. Based on your awareness of the guidelines, you know the [color contrast ratio](https://contrastchecker.com/) between the text and the background needs to be a certain number and that you need a tool to test this.

**Stage 3: Practice**— Feeling more confident in your knowledge of digital accessibility rules and best practices, you use a [tool to measure the color contrast ratio](https://developer.paciellogroup.com/resources/contrastanalyser/) between the text and the background. Then based on the output of the tool, you modify the hex code to meet the color contrast ratio guidelines and retest to confirm you have met the accessibility requirements for this issue.

**Stage 4: Understanding**— You understand that the accessibility guidelines and tools are created with people in mind, and that code is secondary to all of that. One is the means, and the other is the end. In the color contrast example, you understand that people with low-vision or colorblindness need these color contrast changes in order to actually *see* the words on your web page.

This is a bit of an oversimplification of the process. But I hope you get the gist — that there are different stages of digital accessibility knowledge and understanding. True beginners may not be to even stage one, but I am finding that group rarer and rarer these days. The word about digital accessibility seems to be out! Which is great; but that’s only the first hurdle. What I’m seeing now is that a lot of people stop at *Stage 2: Knowledge* or *Stage 3: Practice*— where you are aware of the digital accessibility guidelines, have some testing tools in your back pocket, and know how to fix some of the issues reported, but haven’t quite connected the dots to the humans they impact.

From the standpoint of getting daily stuff done, stages two and three are okay stopping points. But what happens when the things you need to do are too complex for a quick fix, or you have no buy-in from your peers or management? I feel that once we get to *Stage 4: Understanding*, and really get *why* these kinds of changes are needed, people will be more motivated to make those changes regardless of the challenges involved. When you arrive at stage four, you have gone beyond knowing the basic rules, testing, and coding. You recognize that digital accessibility is not just a “nice to have” but a “must have” and it becomes about quality of life for *real* people. This is digital inclusion. This is something you can’t unsee, you can’t unlearn, and you can’t ignore.

**Making digital accessibility a priority — not a requirement**

In my role as an accessibility trainer, I like to kick-off each session with the question: “What are you hoping to learn today about digital accessibility?” I ask this question to establish a rapport with the audience and to understand where everyone is in their accessibility journey, but I am also evaluating the level of company and individual buy-in too. There is nothing worse than showing up to teach a group that does not care to be taught. If I hear the words “I am only here because I have to be” — I know it will be an uphill battle to get them anywhere close to *Stage 4: Understanding*, so I mentally regroup and aim for another stage.

In my experience, when companies and their leaders say “Digital accessibility is a requirement,” nine times out of ten there is a motivating factor behind this sweeping declaration (for example, impending litigation, or at least the fear of it). When changes are framed as mandatory and packaged as directives from on high with little additional context, people can be resistant and will find excuses to fight or challenge the declaration, and any change can become an uphill battle. Calling something “mandatory” only speaks to *Stage 1: Awareness*.

By swapping out one word from the original declaration and saying “Digital accessibility is a priority,” companies and their leaders have reframed the conversation with their employees. When changes are framed as “working towards a solution” and discussed openly and collaboratively, people feel like they are part of the process and are more open to embracing change. In the long run, embracing change becomes part of a company’s culture and leads to innovation (and, yes, inclusion) on all levels. Calling something a priority speaks to *Stage 4: Understanding*.

Some of the excuses I often hear from clients for not prioritizing accessibility is that it is too difficult, too costly, and/or too time consuming — but is that really the case? In the same accessibility training, I lead an exercise where we look at a website with an accessibility testing tool and review any issues that came up. With the group’s help we plot out the “impact to user” versus the “remediation effort” on the part of the team. From group to group, while the plots are slightly different, one commonality is that close to 80% of the errors plotted fall into the quadrant of “simple to fix” for the team, but they also fall under “high impact” to the user. Based on this empirical data, I won’t buy the argument from clients who say that accessibility is too difficult and costly and time consuming anymore. It comes down to whether it’s a priority — for each individual and for the company as a whole.

**What will your coding legacy be?**

The [*infinite monkey theorem*](https://en.wikipedia.org/wiki/Infinite_monkey_theorem) states that a monkey hitting keys at random on a typewriter for an infinite amount of time will eventually type any given text, such as the complete works of William Shakespeare. So by that same logic, a programmer hitting keys at random on a computer for an infinite amount of time will almost surely produce a website that is accessible. But where is the thought process? Where is the human element? While all the things we’ve already talked about — awareness, education, and prioritization of accessibility are important steps in making the digital world more inclusive to all — without *intent,*we are just going to keep randomly tapping away at our computers, repeating the same mistakes over and over again. The intent behind the code has to be part of the process, otherwise accessibility is just another task that has no meaning.

Maybe I’m naive, but I’d like to think we’ve come to a point in our society where we want our work lives to have meaning. And that we don’t want to just hear about the positive change that is happening, but want to be part of the change. Digital accessibility is a place where this can happen! Not only does understanding and writing purpose-driven code help people with disabilities in the short-run, I believe strongly that is key to solving the overarching diversity issue in tech in the long-run. Developers who reach *Stage 4: Understanding*, and who prioritize accessible code because they understand it’s fundamentally about *people*, will also be the ones who help create and cultivate an inclusive environment where people from more diverse backgrounds are also prioritized and accepted in the tech world.

Because when you strip away all the styles, all the mark-up, all the cool features from a website or app — what’s left? People. And honestly, the more I learn about digital accessibility, the more I realize it’s not about the code at all. Digital accessibility is rooted in the user; and, while I ([and countless others](https://a11yproject.com/follow/)) can certainly teach you how to write accessible code, and build you tools, patterns, and libraries to use, I realize we can’t teach you to care. That is a choice you have to make yourself. So think for a moment — what are you leaving the next generation of developers with all that inaccessible code you haven’t given much thought to? Is it the coding legacy you really want to leave? I challenge you to do better for my daughter, her peers, and for the countless others who are not fully represented in the tech community today.

#### About the Author

### **Carie Fisher**

[Carie Fisher](https://cariefisher.com) is an author, public speaker, and developer who is passionate about front-end development, accessibility, and promoting diversity in the tech world. In her spare time, Carie likes to travel, garden, and engage with the accessibility and open source dev communities.