MICHAEL BERGER: Hello. I'm Michael Berger. I lead quality and accessibility over at Basecamp. We make a popular platform for communication and project management, which also goes by the name Basecamp. And essentially, it's a great place to share to-dos and files and exchange messages for group chat when you're running a project or running a remote team, like we do.

So today I'll be talking about how I've helped us become a company who cares about accessibility from the seat of my role in quality assurance. We're going to talk about the idea of accessibility-first QA, how to audit your app for accessibility in a passive way, determining when it's really important to get involved with a project at the very beginning, and helping your team become more self-sufficient.

For a little background, I joined the company way back in 2009. That's a photo of me back then, actually. And I was part of our customer support team at the time. And in that role, I saw a lot of emails coming in with issues that I felt like we could address before shipping features to customers. And out of that came our QA department.

Now, over the years, we've built a number of different software products for the web and mobile. And today, our efforts are focused on Basecamp 3 and HEY, our soon-to-be-released email app.

Well, three years after shipping the latest version of Basecamp, Basecamp 3, I was searching for a new project to fill these typical ebbs and flows that exist in our QA testing process, and at that, a project which would be meaningful for our company and for myself as an individual. I had just experienced watching my grandmother, who was an avid reader, struggle with vision loss, and it really showed me what life with a disability was like for the first time in a way that I had never seen before. So that's really what motivated me to sort of dig into this work.

Now, before that, it's not like we didn't realize accessibility was a thing. In fact, we would, back in the day, get emails from customers asking about whether our products were compatible with screen readers. And oftentimes, our response at that point was that we weren't really sure and we just sort of suggested giving it a try. And as a small team, building software to scratch our own itch, accessibility just wasn't a priority at that time.

So when I decided that I was going to give accessibility a go on my own here, first thing I did was run through a voiceover tutorial, and then I flipped on my screen reader, loaded up

Basecamp, started navigating through elements, and immediately started to cringe. The experience was pretty bad. It was pretty unclear what you were doing in the app. Many controls were inaccessible.

And so I did some research. And while I don't have a strong background in coding, I was able to open up the Basecamp code base in my text editor, and I just started making some simple tweaks, mostly adding aria-labels and hiding redundant elements. And then retested the app and discovered that these simple changes I was making made a real difference for the user experience. And it created this positive feedback loop for me that just made me want to keep on going.

Now, to give you some background, the sort of QA that we do here at Basecamp is oftentimes referred to as black box testing. And to put it simply, it means that we test the product as a customer would. As I started digging into accessibility, I kind of discovered this idea of accessibility-first QA. I felt like I could enhance the QA I was already doing by using a screen reader and keyboard navigation to drive the exploratory testing process. We would still find all the bugs and usability concerns we would before, but in addition to that, a whole new class of issues that people using a screen reader and keyboard navigation would experience. And it gets even better when you consider that a more accessible app most always results in better usability for everyone, all customers.

Now, all of a sudden I was finding a ton of accessibility problems in the app. But it's not really possible to handle issues outside of the project's scope. When we're doing QA on a feature, we're really looking just to fix bugs associated with that feature. And at the same time, I knew that eventually meeting WCAG 2.1 accessibility would be a goal of ours. So I had to figure out a way to passively document pre-existing accessibility concerns outside of just dropping everything and doing a full-on audit.

So the way that we typically handle things like these sorts of situations in Basecamp is by creating a Basecamp project. So that's exactly what I did. I spun up a project for accessibility. And I just started throwing some to-dos as I was finding them into this project. And eventually, I saw patterns and I was able to start organizing these findings into groups, like places where autocomplete pickers didn't work with screen readers.

Here's a detailed look at one of those to-do reports-- describing the issue, the expected behaviors, along with links to examples in the app. Other similar groups I was able to create

for communicating alerts and using aria-live and modal dialogs that didn't properly trap focus within Basecamp. And eventually, I was able to create a higher-level list of priorities that sort of lays out what I see as the most important things to address in Basecamp in order for the app to be accessible.

Now, sometimes there are projects where it's especially important for QA to get involved early on. I'd like to show you one of these projects and how I approached researching how to make the design and interactions accessible.

So this is an example of a hill chart. And it shows the progress of to-do lists on the visual bell curve. To describe what we're looking at here, we have a bell curve, the left side of which is labeled Figuring It Out. And this represents the problem solving, uncertainty, and unknown stage of the project. And the right side is labeled Making It Happen. And this represents the execution phase.

And so the way that it works is you put the screen into an edit state, and you drag each dot to its new place along that curve, and then you can add an optional note. And at the beginning, for example, all of the to-dos, all of these to-do lists are going to be stacked at the very left side of this hill chart. At the end of the project, everything is going to be stacked on the right.

Now before I move on, I'd like to give you just a few seconds to sort of think about what possible accessibility concerns there might be around this design.

Just to further clarify this concept, here's a look at the hill chart history, or progress view. What we see here is a list of updates over time, and it can show you sort of how progress is moving along in the project.

So when I start thinking about how to make an interface accessible, the first thing I typically do is turn to the W3C, which is the World Wide Consortium's page of example patterns. And I look for a pattern that seems similar to the interface that we're designing. And in this case, I felt like a slider seemed most similar to what we were creating. When we dig into the slider there, we get an explanation of the keyboard interactions that the widget is expected to have, along with all the markup required to achieve what's demonstrated by the example.

And my favorite part are these functional examples that they provide. You can interact with these using your screen reader to understand firsthand how they function, and then you can later compare these behaviors on these examples to your final design to make sure that they

work similarly. These can be especially helpful when the widget you've designed is misbehaving, like on a particular platform, like say iOS or Android. You can go back and retest, and you can see if even the W3C example works on those platforms. And it sort of gives you some guidance on how to proceed with troubleshooting.

So once I did this investigation, I took all of this into consideration and put together a document of recommendations for the development team. This covers each different view of the hill chart. And, for example, the to-do list per month, this is one of three different views for the hill chart. And for each of these different views, I've detailed the interaction and navigation with example screen reader announcement text for both the view and the edit states, as well as any outstanding questions that I had about how it works.

So these are hill charts. And this was a project that seemed very complex for accessibility from the onset, but which became clear and very actionable once I broke it down.

Now, our QA team is pretty small compared to the rest of the company. Internally, it's just me. And in the last six months or so, we've supplemented my efforts with the help of Aspiritech, which is a great local organization who employs testers on the autism spectrum.

Projects at Basecamp are broken up into six-week cycles. Now, some projects can span that whole six weeks. Others take just a part of that. And these development teams are comprised of one designer and typically one to two programmers on the project.

Once the development team feels confident in the design, they bring QA in to evaluate. And we go ahead and run through the feature, exercising it aggressively and searching of functional bugs, usability concerns, and accessibility issues. And then these findings are documented as to-dos in Basecamp, of course.

Now, a good portion of the accessibility issues can actually be uncovered using automated tools during development. And encouraging designers to run these checkers, like the axe extension, can be a really big time-saver, in contrast to QA documenting every little glitch-- for example, on mobile screens, a bunch of elements being under the minimum touch size recommendations. Making to-dos for each of these elements could use more time than it would take a designer to simply find them themselves and fix them. And by shifting the load of that work, it provides QA with more time to dig into aspects of accessibility that can't be determined using automated tools. We can design a non-visual interface for customers who use a screen reader, for example.

So to aid in this effort of getting designers and developers to be self-sufficient, I created an accessibility wiki. And it covers techniques for evaluating accessibility across all the platforms that we serve and includes links to a lot of useful resources that I've found helpful.

Just to show you a few examples that I've added to the wiki, we have here a reminder to not ever hide focus outlines. We have techniques for using non-visual spans to sprinkle in additional context for people using a screen reader. And one of the many important ones-- how to use a browser extension to find common issues. Currently, this wiki has some screenshots and references to an unreleased product, but I plan to open it up to people outside of Basecamp once our new product ships later this year.

So as I mentioned earlier, most of our company is remote. And one of the important ways that we communicate at Basecamp is with team kickoffs and heartbeats, which we share at the beginning and end of each cycle. A few months ago, I decided that it would be a good idea to split out the heartbeats from my team into two, with one focused on QA and another focused on accessibility, just to sort of bring more visibility to accessibility.

So in this example heartbeat, I've talked about some new resources I've discovered. I mentioned a tidbit about the discovery that too much contrast can be a bad thing and why title attributes are a big no-no. I detailed a method that I used to make an interaction more accessible within the previous cycle, and additionally, a recap for an accessibility meetup that I went to.

Now I'm not sure that everyone at the company reads the completion of these posts. But even so, I think that seeing it come across their dashboard offers a reminder of this work and why it's important.

So three years in, every feature we've shipped since 2017 has met a baseline for accessibility. And that's just a major accomplishment for us. And in addition, our brand new app HEY, which we'll be launching later this year, will be our most accessible product to date.

And final note is really to have patience with yourself when you're taking on this work. For a long time I was frustrated that we weren't making progress as quickly as I wanted, and I felt like I was sort of carrying the burden of accessibility on my shoulders. But what I've learned is that it may take time for your company to catch on, and being persistent, reminding people why this work matters is really all that you can do, and to work on being OK with that and being

patient with yourself.

And finally, I'll end with some resources that I've found most helpful over the years. The accessibility Slack account is super useful for tough questions. If I've done research trying to understand how to make an interface accessible or trying to debug why something doesn't work in a certain browser and I just can't find a solution, I'll jump over to the accessibility Slack account, post it there, and within hours I tend to get a really wonderful response.

Twitter is another really great resource. And I threw together a list that I shared here of all the individuals and the accessibility community that I personally follow. Meetups and conferences have been a big part of my development. The Chicago accessibility meetup has obviously now shifted to an all-virtual setup. And so if you're not even in Chicago, feel free to join us sometime, or maybe even start a meetup group of your own.

And finally, read as much as you can about accessibility. Laura Carlson's newsletter is a great place to start. But really, anything that exposes you to more about the world of accessibility is important.

Well, thank you for joining me today. I am always here and happy to chat about accessibility. And I just want to thank you all for coming.