Design Notes

with Will Larche

📻 Listen

Liam Spradlin: Will, welcome to Design Notes.

Will Larche: Thank you, Liam. I'm so excited to be here. Finally. You took your sweet time making me a guest, didn't you? But I appreciate that you wanted it to be the biggest hit that it possibly could be before you even approached my people, and that's kind.

Liam: Right. This is long overdue and this is part of the reason that I'm loving the M10 campaign so far because it does give you an excuse to reach out to Will's people and finally get some of my favorite coworkers on the show.

Will: I will admit there are some people that are wondering what happened to M4 through nine, but we'll explain. We'll make it-

Liam: Yeah. Don't worry about it. Okay. Will, for the listeners who don't already know, who are you and what are you working on?

Will: I ask myself that every day Liam. I am Will. I am an engineering manager here at Google. I am now the lead in Google design platform for AI for designers and developers.

Liam: Okay. As soon as you say that there are going to be people wondering what that means.

Will: So I manage global cross-functional teams that attract and retain users to the Google ecosystem through improving its look and feel and save the company money by increasing the productivity, efficiency and creativity of designers and developers through the application of large language models to common problems.

Liam: I want to know first of all about the journey that led you there from the beginning.

Will: What beginning? Birth? Where are we going?

Liam: You should start where you see the start.

Will: It started for me when I joined a Buddhist meditation group in the West Village called Friends of the Western Buddhist Order, now called Triratna. And their leader did IT for a Georgian economic publisher. If you're not familiar with Georgian economic theory, I'm not here to espouse it or promote it, but it's a theory of taxes based on property owning instead of income. And he was going on vacation and knew that I was computer-literate. Growing up in the '90s you had to learn how to take care of a computer. You couldn't just have one that worked. That not was not the goal of the companies that were making computers back then. It was more like a fun hobby that you were always building your computer. And he said, "Can you come in twice a week and restart the printer for the bookkeeper?" I was like, "Yeah, I can do that."

And so then I did that. And he sent me this message while he was away saying, "We need to change the copy on the website for the charity." And I was like, "Okay, I'll try that." And I went into Dreamweaver. If everybody remembers Dreamweaver. I believe you can get me through the night. It was a simple task. Change a sentence on the website. I could not do it. I could not figure out how to do it. I did a poor job. It made me mad. I went home. I was super upset and I said, "Maybe I'm blaming Dreamweaver. Maybe the real person to blame is Will because Will didn't know anything about this." And I said, "If I knew about this, if I understood what I was doing, would I like this?" And the answer was yes.

I was pretty sure that I really would like working with website as I thought at the time. And so then I decided to get some education. I ended up taking a class at NYU night school called Intro to C which was like, what's a compiler and how do you use the plus and minus operators? And from there, I talked my way into an internship telling them I wanted to be an iOS developer because iPhones were new at the time and I was super excited about how nice they were and how beautiful they were. I wanted to make apps and I wanted to understand it. I got that internship, it turned into a job. I then got another job. I rose at that startup to head of mobile development. Then I ended up becoming a VP of engineering somewhere, transferred around to a lot of other startups and agencies. I was chief product officer and mobile lead at the same time that

Google was reaching out to me and I said, "No, Google, I'm not interested. I don't know why you'd think I'd want to work for you. I had no interest in being a cog in a machine." But then my startup closed and I said, "Hey Google, let's talk."

And then I actually turned down several teams that were interested in having me work on them because they just didn't feel right. And then they said, "Well, have you heard of Material Design?" And I said, "Heard of Material Design?" When I was managing designers back at that startup, I didn't really know what I was doing, but I had one that created her first mobile designs ever and they didn't really work. They didn't make any sense. And I had seen Material Design come about these external guidelines and I said to her, look at these and see if they could help give you some knowledge or inspiration. She did. Two days later, gorgeous mobile designs that were usable and buildable that turned into our product. So I had already seen the power of Material Design to help someone fundamentally and change essentially the path of a product from being uh oh to wow. So yeah, I was excited to work on Material Design.

I came in as an iOS engineer and motion expert, and then I ended up founding our Flutter team that ran that for years. I ended up running our color team and our color space, the HCT color space, Material Color Utilities, our work in personalization. And then my engineering director asked me to start an AI initiative, and that's turned into a bit of a thing. I've also built Figma plug-in tooling. I think I sent you a list of all the things that I've worked on. It's actually a lot, I think for one person. I bounced around, but always within design engineering here at Google. An now I'm here today.

Liam: To be honest, I didn't even need the list. I feel like the stuff that you've done has had such a big impact on the team and by extension all the people building with Material that it's impossible not to know the list.

Will: That is so kind.

Liam: So something between Flutter, the HCT color space, working on Figma tooling, also this perspective that you have about making things more efficient for the people who are making software. Something that I really want to ask about is the creative nature of making software from an engineering perspective, because your work intersects so often with design, and I'm curious what your perspective is on the relationship between those two things.

Will: It's really weird. So engineering is the study of choices. We have the thing we need to build and there's 10 to a hundred different ways to do it. And some of them are obvious, some of them are more avant-garde. Hey, you want to go write everything in machine code, fine. Or you can use the newest framework that hasn't been battle tested yet, but you have a good feeling about. Those are the choices that engineers have to make every day. So there's something creative to that. And then once we've made our choices, we sit down and we write. So it's very much like being a novelist or some other kind of writer except that there's a right answer in the end that you can write whatever you want, but it is always supposed to do something that's been predetermined. We've gotten requirements, we have a spec, we have designs maybe if we're doing front-end work and it's supposed to look like that and go like that.

So there's all this creativity towards a constraint, which I find very different from visual design. When I watch you all work, I've often seen you all do projects where it's like, "Hey, what's next?" Where it's literally just like we should be thinking about the future and what does that mean? And then Karen Ng, one of my favorite designers goes, "I'm really interested in mycelium, the roots of mushrooms right now." And people go, "Wow." And then that turns into sparkles and Material 3 and stuff like that. It's a totally weird process where you all don't have a right answer. You have to just create and do stuff, and it's very open-ended. And that's a little intimidating to me. That's why I consider myself not at all a designer in any way, but it's design fan. I am here because I love what you all do so much and I want to make it come true for the user and I want to give them a high-quality experience because of the amazing ideas that you all have, the magic that you pull out of the air.

Liam: Okay. A couple interesting things there. First, Karen is absolutely right. Everyone should be paying more attention to mycelium.

Will: Yes.

Liam: And second, it's interesting to hear your perspective on that because I always think about ... When I talk to people of all kinds of different creative disciplines, I'm thinking about what is the material that they have to work with, how do they manipulate it and what are the constraints that are placed on it? And I think for me, even talking about some of this visioning work or discovering a new vibe or coming to some conclusion about how the interface should work in the future, even there, it seems a little bit more clear to me, but I am really interested in this comparison you made between writing code and writing a novel. How do the materials and the constraints work together when you've made your choices about how you're going to approach it and now you're writing and then what happens?

Will: Well, you are making the assumption that we've made our choices, and I think most of the time that's true. Sometimes there's just pure exploratory software engineering, but oftentimes it's about the fun of it. So I guess knowing what I know about all the arts, you're manipulating whatever you have either until it does the thing that it's supposed to do or it gives the feeling, which I assume is what you're probably doing in the visual arts. Is that you just have this intuition and feeling that it's done. Yeah. Because that's what happens in my music, in my theater work too is, which I haven't mentioned yet, but this is all part of a big ruse for me to be able to do a queer avant-garde music theater in the East Village that's very expensive and so thank you Google for supporting me. It's difficult sometimes because it can seem like a sea of I don't know what to do and it could be anything.

And then I've also run into where I've made stuff and people are like, "Why don't you just keep it? That's good enough." And I'm like, "I don't know how to tell you that it's wrong still. This still isn't the right thing and I need to tear it down and build it again." And it can be very frustrating for other people to see a writer do that. So engineering has an appeal to me in the sense that I live in this world in the arts that's total ambiguity and is so difficult, and then I can come into work and there is a right answer, but still requires some creativity on our side. I think as time goes by in the industry and there's more constraints based off of resourcing and we're not just in hyper growth mode anymore, we do actually expect people to make more choices ahead of time and to go ahead and build the thing the right way the first time.

But I'm smart enough as a manager to know that there is no such thing as the right way. We need to revisit things. We need to do post mortems on work all the time. We need to refactor, we need to build in time to change things and then align that to when business priority shifts so that we are making the next version of the thing when it can also have a difference for the user. And so as a manager, I'm often trying to enable my people to be able to do their creative work, to learn from it and to improve things while also intersecting that with what the business needs.

Liam: Yeah. There's an interesting recognition there that what is the right thing is so contingent on the rest of the context around it. In engineering, in design, certainly in writing, if I'm writing something and I'm working on a draft over multiple days and then I come back, I can definitely tell what mood I was in when I was last working on it.

Will: There's also this phenomenon in engineering where every engineer thinks the other engineer is wrong. It's very weird. And I think that's a function of our human brains where if one engineer is working on something large, they'll spend so much time putting themselves into it and like I said, dealing with thousands of choices, every single thing that they decide to write, the way that they name a variable to whether or not they're using an if statement or a switch ... Sometimes these things are predetermined, but most of the time it's up to the engineer to decide. These are the very personal pieces of engineering that then when somebody else comes in ... It's like when you're in somebody else's house and you might most of the time feel like, "Oh, I love this house, but I would do it different." Not that rug. That sort of thing. Engineers have the same problem. Where even if they like something, they're like, "Well, I would've done it differently."

And I think one of the mistakes engineers make is thinking that that feeling is truth that has to be followed all the time instead of finding a way to learn more from other people. The best engineers are the ones that are willing to look at what's going on, learn from it, and then figure out what they would do next, not what they would've just done in the past. That's useful in a post-mortem or something, but most of the time we're trying to move forward.

Liam: Yeah. It is interesting to make the comparison to how someone has decorated their house because at least for myself, I certainly think of code as something more objective than design. So it's interesting to hear how big a part of it actually is, the subjective input of the person writing it and all these small decisions that ... I feel like in design, I really have to be conscious about keeping up with that so that at the end I can understand and also convey that context to someone else in a way that it makes sense.

Will: It's definitely underappreciated the million little choices inside the code and how the person who was writing it felt that it was beautiful or still hated what they were working on. I don't know if designers run into this, but I think most people when they're working on things can be like, "I hate this, but I have to move forward." Sometimes that's actually when you do your best work though, is when you have to accept that you couldn't get something to totally please you or to make it perfect, but then you're able to ship and have an impact.

Liam: Yeah. Disappointment can be focusing.

Will: Yeah. And you know what? Disappointment's played a very big role in my life, I'll tell you that much.

Liam: Do you want to talk about that?

Will: No, I don't. Liam. That's very personal. Why would you even bring it up. On the house decorating thing though you've been to my house. You know how I go. I'm always aiming for

... Since I'm such a fan of design, but I don't have any talent for it, so I hire really great designers and tell them to go nuts and the feeling that I want is someone walking into my house and going, “oh wow, that's a little much.”

Liam: That's great.

Will: That's me. I'm glad my husband feels the same way.

Liam: It's a good spot to aim for. I want to go back. I don't want to brush under the rug all the stuff that you were saying about your work in musical theater and writing. I've been in the audience at some of your shows. I want to talk a little bit about that and I also want to talk about how the interplay between that and the work that you do day to day. What does that feel like for you?

Will: Well, it feels like I'm acting like an engineer. I deal with my psychoanalyst. He always tries to tell me, do you feel like you're performing? Yes. Yes, I do. All the time. Doesn't mean I'm just ingenuous. It just means that I do feel like there's a heightened sense of being aware of how others are perceiving you that is very valuable if you can wield it. And I think one of the reasons why I've had exactly the trajectory career that I've had is because I'm willing to bring personality and a curiosity about people and empathy and essentially my performative skills to something that can be dry. Software engineering can be a little bit like, "Hey, just leave them alone. Let them go stay in their room. Don't bother them."

What I really, really want is for people to understand each other and to always come at whatever's going on with a maybe attitude. Whenever designers, product people and marketers, whatever, come to an engineer, the engineer needs to say, "Maybe." Listen to it, evaluate it, and give them the time to see that. It might still be no, but don't start with no. That's such a way for good ideas to get lost and for amazing things to never get built, unfortunately. But it is like this tradition in software engineering that we are crusty and that we put up a front that tells people, leave me alone. We actually don't do our best work that way. We do our most self-indulgent work that way because then we are just writing for ourselves. We are collaborators. There's clients and there's end users, and all those people are the responsibility of the engineer as well.

Liam: Yeah. I think design and engineering have shared goals around those audiences, and I also think we are all in one of the best positions we can be in to make these things possible I think.

Will: What do you mean?

Liam: Meaning that we are in a position at Google working on Material Design, lots more resources than in certainly any of my previous jobs to find the solution. I think if we can't find it here, I don't know if it can be found.

Will: I'm going to disagree with you. I think we have far more constraints these days than we appreciate. One of the things that's been on my mind a lot lately is why is that and how can we fix it? Because we do have this responsibility to make sure that with 200,000 people working for us, we are doing great work in moving fast. And we have this way that we've worked for the past 20 years of building shared infrastructure like Material components and the many, many things that we have internally that run our back ends and our front ends. I would just like the people on the outside to know that engineering inside Google is completely different than it is anywhere else because we have custom shared infrastructure. And every time somebody builds something like that, any little thing that we're building that's supposed to be shared, we're making this promise that we're going to help the teams that are using it move faster.

But I think it's actually worth asking the question as to whether or not they do. If we want to change something, we end up breaking somebody else's application that's doing it, or we have to talk to a hundred different teams and sell them on what we want to do, and what if only some of them do, and then we have to align with more people. It becomes harder and harder to move forward every day. There's also this tension in Material Design between are we building blocks or are we creative direction? And we've always agreed that we're both. Actually when we started right before I started here, when earlier teams were releasing the first Material, there wasn't components at all. It was no building blocks, it was just creative direction, and we changed the world that way. But then people kept saying, "Well, you want me to have a button that looks like this, give it to me." And it seemed reasonable. And my team, when I started on the iOS team here, it was actually people inside the company building a button being like, "Hey, I'm on Maps and I built this button. Would you like to use it on search?" And then somebody being like, "Yeah, put that somewhere where I can copy it." And then that turned into a directory of Material components that people had made.

And then Material was like, "Should we help with that?" And people were like, "Yeah, it seems reasonable that we should help with that." And then over time it became creative direction can't change without the components being ready. And we released these things together and people keep asking for that. That's what they expect. But it might be that that is still missing the mark in some way in being able to bring true innovation. And I wonder what it would look like if we had a separate team for creative direction and a separate team for building blocks. And the building blocks were really, really good at being flexible and they could build custom UI much more easily. Because secret, even in a Material app, only 40% of the UI is Material and then the rest is custom to the content of that application. And so I think that in a better world, we would separate these things and if we had components, they would be so flexible and divorced from their styling that we could change them at any time. So we really should be building things in a way that they're so flexible that we are indemnified for a future we cannot see instead of calcifying the spec of today into components that are difficult to change.

Liam: Yeah. And I do want to talk more explicitly about where Material was when you joined and where it is now and how we got there. But I do think there's a really good point there that we progressed from or emerged from a collaborative, fast moving, chaotic way of assembling designs into something highly systematized in which our concept and perhaps others' concept of how to implement a design system was along these very objective lines of components that look exactly right, the shadow values are exactly how they should be, the size, the typography, everything. Great. And that now we're moving with relational color and all of those things into a space where we're trying to find a way within those boundaries to, as you said, communicate the creative direction or communicate more of a mood, more of the subjective quality of the design, and potentially facing a future where we have to build those flexible components that maybe we should have built from the beginning. But I want to talk a little bit more about how you've seen that progression from where you started in concrete terms and then we can extend into the future.

Will: Well, I think the biggest change was it used to be designers were just making guidance and then everything else hopefully happened. So this was this upside down world where design was completely in charge. They were still not having a good collaboration with engineering as is so often the case, but when the design system is the product they could just go and do. And I think that's rare. I can't think of many other products where the engineering is second maybe or not the most important thing. The components were not always seen as necessary. It was our clients that told us they were necessary because it wasn't going to come true for most of them unless we supplied components that became the pattern. And it seemed like I said this reasonable expectation that we should always have components that match the guidelines.

So eventually we got to this point where they would tell us that the guidelines were changing or had changed, and then we got to this point where we said, "Hey, stop releasing guidelines until the engineering is ready." And then these things became ... They marched in lockstep. Again what happened under the hood that was more difficult is the engineering is then the blocker, not for being built, but for being shipped because we have hundreds of clients with inside our own company, much less on the outside and every time you change something, people complain. They've maybe customized it in some way and you didn't know it, and then they're like, "Hey, I was using this button as a date picker and now it's not working." And you'd be like, "Oh. Well, at Google we have this idea of a monorepo, which is one place where our code is stored, and it would be our responsibility to resolve that problem even though somebody was using off-script behavior."

And same thing would happen on the outside where people would be like, "Give us the next Material. We're so desperate for it. Why are you holding it back? Why are you so slow?" But then if we did it in a way that we were changing what they had an even larger contingent of, people would come forward and say, "You just broke my app. I did not ask for Material 3. My app doesn't even look like Material 2, but I built it off of Material 2 to do this other weird stuff." And since we had done it in this way that wasn't really built for flexibility or flexibility was always something that was just figured out locally that then it became impossible to move forward that way. And I think that tension between are we building blocks or are we creative direction?

That's the largest conflict that I see still going on inside Material Design. Some people think we still don't have enough people to really even do what we're trying to do, which is to power the entire expression of Google brand through software as well as enable people in Android to do the same thing for all of their brands or to use a built-in baseline that they can theme on top of that. And so on top of that creative direction to also just be how you build apps. To standardize that we are the button, we are the text field and that sort of thing. And each of those components is a tiny application. That's huge. Some of them are huge.

I worked on text field on iOS ... I once printed it out because somebody was like, "I don't think you're technical enough to get to the promotion you want." And I was like, "Hold my beer." And then I went and printed out all the code that I had written for text field, had it bound and dropped it in his lap and said, "I've written this much code. Did you know that?" And they didn't because they thought it was just text field. And I was like, "This is what it takes to do a text field. It's ridiculous." And that was even back in the day when I don't even think I did my best work or we had the same standards that we do now where it would be even better, even stronger now for looking exactly like the way it was. Material was scrappier back in the day too. Like I said, it started with, I made some code, can you share it and maintain it? Sure, we can do that for you to, my first project was actually building the IO app for some reason. We were like, "Okay, let's do that." Now we're like, "No, we do not have time to work on the IO app." We power the whole world of front end in many ways.

So it's been quite a journey and I'm still very, very proud of some of the things that have come through. Dynamic color, user selected contrast, the HCT color space, I think those are the greatest things that I've touched here because they are truly innovative under the hood. They use some of the most bleeding edge technology for dealing with the color, and they brought new concepts that hadn't existed before in order to personalize UI. And every bit of research that we have shows that people want personalized experiences, and that's what Google's known for. We personalize your search results. We're going to be personalizing your UI more. I think that's what my AI team is going to be working on.

Liam: Yeah. And people who have followed me on my Google journey know that I was saying years before I joined this company that this was the place best positioned to realize a vision of true adaptive design. And I really see making these subsystems relational in a way that abstracts them and allows them to respond to unpredictable conditions is how we reach that future.

Will: And I also want you to ask yourself, why haven't we reached the future already? It's Google. Google is gigantic. Why have we still not gotten to a fully adaptive software for all of the things? It's because it's still expensive. We can make helper code to help people get a tablet design, but then Gmail or whoever these giant applications have to sit and design it and make it and they have their own constraints and their own concerns. I think it's good for us showing people a way to move forward, but then they need to take it on board and then apply it to their thing.

Liam: I'm thinking about the dimensionality of the interface that we're talking about, two dimensions, something behind glass that you can't even physically touch and how you work more complexity into that system at a level of abstraction that's appropriate for millions or billions of people.

Will: When I started on iOS, it was about how do we make this look like the real thing? The skeuomorphism of it all was so much fun for engineers to work on, and I hope designers too, where you'd be like, "This is a podcast app, so it's got to look like a podcast studio with the microphone and all these sorts of things." I miss the fun of that, but I don't miss the look of that, if that makes sense.

Liam: But I have to say, it's interesting to me to think back on that now because I think metaphor has been at the heart of translating interface into a 2D screen since the beginning. And it's no coincidence that the first creative conceit for a desktop interface was called a desktop. I can't help but think that that's going to re-enter the picture at some point. The level of complexity and personalization that we're talking about, we'll have to take on a quality that helps people understand these concepts by referencing something that they already know, which I guess was the conceit of Material.

Will: Yes, it was. And that was really cool. Although I think we actually did help ourselves by expanding that universe a little bit and stopping to be so literal about the Material paper. But I see in the future where this is going to come into play. Working on the AI design team, we're working with teams like Bespoke, which is this team that has been featured in Gemini's press about how we can build designs at runtime that are completely bespoke to the user. Take this to an nth degree where you are seeing a UI that has been sending you experiments for years to figure out exactly what you like and exactly what you need, because everybody also has very different usability issues. One of the things that we learned here is that some things like WCAG are actually constraining and not taking into account the fact that there's huge diversity in usability needs inside the world, but we could begin serving that to you. We could figure out through your usage exactly what you need and build you a UI that looks nothing like mine.

Your experience with an application, the same application could end up becoming strange and weird and take into account the indeterminacy of LLM work turning up the temperature if you want to see something that's avant-garde almost but works for you. And so we could end up with concepts that we don't even have yet that the computer tried out, that then we could codify into guidance that we could use in other places because if something works for another person, it could be a good experiment for somebody else.

Liam: This is why I came to Google.

Will: And then ask yourself, what is the value of the design system in that world? As we march towards that, the design system is going to ... Sure, we're going to start with just where are the components on the page? Then we're going to start to, well, is it these components? Is design guidance about patterns, usability, creative direction and expressing brand so that people still have a halo feeling of Google made me feel happy, and so I care about it while I'm doing something that's pink and completely different from other people? And our design system's going to look the same way that they do in the future. Would they maybe for an end user product like an external company, but not for Android or Chrome or something. There's these huge possibilities that design systems blow up a little bit and change that. If we give control over to the computer to begin making design decisions, then we have to just think about how we're guiding the computer and not guiding other designers necessarily.

Who knows? This is all theoretical right now. Fun experiments are happening and it would have to be proven that it has value, but it's a great thought experiment to just think about if I took away components, what is the design system? I think also about UI from science fiction, from Star Trek Next Generation, and they had touch screens with all this stuff and all those beautiful jewel tone colors and how I think on purpose it looked nonsensical to us so that we wouldn't spend all of our time gazing into it instead staying with the people.

But also it seems like things that went further from the metaphor. It went further away from the desktop like you were talking about. Maybe for some people things will iterate to that place. We've done a lot of research here that showed that applications in China look completely different than applications here. The people wanted more density, more things, whereas us having to do with Latin and romantic languages need air around our letters and space in order for us to consume things correctly. So we have that Italian influenced minimalist design in a lot of Western UI that then doesn't speak to the rest of the world. If we gave these experiments and let the computers do things, we might end up with new paradigms, new components, new things that just haven't been explored yet, and then we could end up with that Star Trek design because it organically happened and worked for somebody, which I think is really cool.

Liam: Yeah. And I think what's interesting to me about that, about seeing an interface that's so abstract that someone is using it perfectly successfully, and I have no idea what's going on, it just looks like shapes on a screen to me, for some reason I'm picturing a pink screen with an orange star in a green circle or something.

Will: That's because you're looking at me right now. I think that's what I look like.

Liam: But I think what's interesting there is that there must be a recognition not only of what we're responding and adapting to, but the ways that those responsive patterns also affect the environment and the context in which they appear that there's a subjective impact on the person using it as well that I think is going to be really crucial as we carry out these experiments and see what direction we want to go.

Will: I miss putting in detail Easter egg love and attention into all UI the way that we did when iOS first came out, and that was kind of the standard that happened. As applications were becoming a thing and people found that the way to get noticed was by doing that sort of thing and before the markets just decided, oh, it's these five apps that you use on a daily basis. And when things were skeuomorphic there was so much opportunity for that. It was like, "Oh, I could make something that when the billiard ball goes across the table, you can almost see the hair of the felt change." And we all loved that sort of thing. Or a pull down ... Remember the pull to refreshes movement that had wacky things? It's a pizza app, so then it's a pizza being built, coming together and then being eaten while we're refreshing your UI. I miss being able to put that sort of level of love into end applications. I don't work on end applications anyway, but I feel like there's still great opportunity for us to bring that detail and love to users and that they appreciate it.

Liam: Yes. You can tell when someone was happy making the thing that you're seeing.

Will: Well that's why you don't cry while you're cooking because then people will get sad when they eat it.

Liam: True. I have to say, going back to my point about my exuberant optimism about Material and where we're positioned, there was a part of me that couldn't believe that we shipped the wavy progress indicator for playing music on Android. That that became a system component. I think we are getting to a place where the joy of these components is allowed to be expressed, and to be able to do that at a place like Google with all of the complexity that you talked about is pretty impressive I think.

Will: I'm going to give some props to my friends on the Android team, Lucas Dupin and his team worked on that slider because Android definitely has a culture of wanting to go that extra mile sometimes. Dan Sandler, one of the directors of engineering over there, he's responsible for the Easter egg that gets put into every single Android release, which is so cool. And so Lucas was showing me the slider as he was working on it to put it in there, and I don't believe anybody asked him to do that, but he saw it and loved it and the creative direction was so strong that he said, "This feels incomplete until I have it." And that's the great partner that we love to work with over here is when that matters to them. That we're not like, "Trust us, this is great," but instead they're like, "Oh, I see that it's great and I can't wait to give this to people." And there's also a tradition in Material from several Material releases of us showing people example applications of the creative direction that usually involve the music app. And then we have been called out on Twitter before for never building the music app. People not understanding that's not our department. So I'm really glad that this was one of those times that could come full circle.

Liam: I have been among those calling us out before I was a Google.

Will: Where's my music app? Where's my Material 2 music app?

Liam: We saw the animations, but they're not on the phone. Where are they?

Will: people don't even really know all the things that actually don't even make it to ever being shown to the public, but they're sometimes incredible. But just get cut for time in the movie or anything. I think I mentioned this to you in something I wrote to you recently. Material 2 was so revolutionarily gorgeous, and they spent so much time coming up with a hundred ideas for it that I used to just sit inside the deck as a lowly little engineer enjoying myself, letting my eyes have a feast. And there was this one thing, image treatments we called it, which was applying filters and ... How would you describe an image treatment?

Liam: Yeah. I guess my understanding is that some of them were like shaders. You're transforming the image visually in a way that could be a duo tone treatment or a halftone where it's composed of dots and the dots are changing size dynamically, mostly loading states or ways.

Will: That's the one. Yes. The dot loaded state one is the one we all remember because it was this idea of showing, I'm uploading it in a chat app because Google has a few of those. I'm uploading this image. It's going to take two seconds so I have this treatment of the image turned into Lichtenstein dots that are changing size to make a wave to show that something is happening before it turns back to full color again to show me that it's finished. And it was pure inspiration. I asked the audience to begin a campaign saying, "Bring the image treatments, Google. Released the image treatments. The director's cut of Material 2 deserves to be seen."

Liam: It does. It absolutely does. I will join that campaign. I'll sign the petition. That's really where you could see the joy of the system coming out.

Will: Absolutely made me feel better by whatever was going on.

Liam: Will, thank you for joining me on Design Notes finally. This has been a fun conversation and I'm glad that we could finally have it on mic.

Will: It's been my pleasure, Liam. You know that you are absolutely one of my favorite people on the planet, so to get to bask in your glory for a moment is really quite an honor so thank you.

Liam: Thank you.