FOLKS REINVENTING THE WORLD ONE OBJECT OR IDEA AT A TIME

Erik Kettenburg
Dave Merrill
Nathan Seidle
Eric Stackpole
Eben Upton
Catarina Mota

MAKERS AT WORK

Laen
Zach Kaplan
Emile Petrone
bunnie Huang
Natan Linder
Ben Heck
Becky Stern
Ward Cunningham
Jeri Ellsworth
Sylvia Todd
Dave Jones
Bre Pettis
Eric Migicovsky
Ian Lesnet
Massimo Banzi

STEVEN OSBORN

FOREWORD BY BRAD FELD

For your convenience Apress has placed some of the front matter material after the index. Please use the Bookmarks and Contents at a Glance links to access them.

Apress®

Contents

Introduction

I am a big fan of the original book in this series, *Founders at Work*. One thing that stuck with me about the stories in *Founders at Work* is the fact that CEOs of massive, successful Internet companies are just regular people who have overcome many obstacles and failures. Great projects, great companies, and great products don't just happen. These things start with one or more people who have the enthusiasm and desire to challenge what everyone else has done before them. These folks also have stories—not just about triumph and achievement—but also about failure, overcoming adversity, and persistence. Failures and challenges are a part of anything we do as humans, and how those failures and challenges are overcome is where the best stories are tucked away.

Stories are things we can all learn from and draw inspiration. I started this project because I have a passion for making things and I wanted to hear and share the stories behind the scenes, from inside the workshops and garages of people leading the way and building interesting projects that inspire other people to catch the maker bug.

What is so great about the stories in this book is the incredible amount of diversity in the projects and passions people have. Remote-controlled submarines, 3D printers, pinball machines, conductive ink, blinking suspenders—these are all projects created by makers whose stories are contained in these pages. As you'll see, the maker movement is vast and diverse. Electrical engineers, software developers, designers, schoolteachers, chefs, hipsters, and hackers—anyone can make things. It's one of the most accessible movements in modern history. The tools and knowledge needed to create objects married with technology is more readily available than ever. Although it may sound like a cliché, the saying, "You are limited only by your imagination"—considering the tools available to the average Joe—takes a heightened meaning.

The concept of the maker movement or maker culture is very simple at a low level. It is just people manipulating everyday things in their own environments. A maker's motivation is often to improve the way they interact with the objects and the world around them. The nature and motivation of the projects aren't any different from the DIY culture of yesteryear. The maker movement is just an acceleration of that culture, thanks to modern manufacturing technologies along with the availability and sharing of information via the Internet.

The availability of this information has given anyone with a little curiosity a way of removing the mysteries and magic of the things around us through learning and exploration.

The long-term impact of the maker movement can't be measured solely by the acceleration in the number of people choosing to create things as a hobby. This acceleration is a product of the movement; it's not the key that will drive great social and technological impact. The wide availability of information through sharing is the cornerstone of the maker movement. The conversation about how things work and how to improve them has an exponential effect on the rate at which technology is explored and advanced.

A better measure of the maker movement would be the rate at which this conversation is unfolding. The open-hardware movement is a good example of a subset of the maker culture that is driven by the availability of information. The same way open software has changed the software landscape forever, open hardware designs will enable a new level of collaboration and experimentation at a global scale.

Another important element that has enabled the explosive growth of maker culture is the availability of components at a drastically reduced cost. The era of the smartphone has fueled cheap components, making technology exceedingly accessible. High demands on smartphone manufacturing have drastically dropped the prices of "embeddable" processors and various electronics components like accelerometers and GPS modules. The availability of new and more powerful microchips and the sinking price of components have made new types of projects viable for the first time. All of the things in our world are rapidly becoming more intelligent, more seamless, more connected.

It's hard to think of the maker movement without mentioning digital manufacturing technologies like 3D printing, CNC milling, and laser cutting. Although these technologies are not extremely recent—the first 3D printer was developed in 1984, the first laser cutter in 1965—the availability of 3D printing and other digital manufacturing methods are becoming more affordable, easier to use, and more accessible to designers, engineers, and hobbyists.

The advances in these technologies, along with the broader availability and adoption, are key for makers. These technologies enable individuals to produce extremely detailed and precise objects that would be difficult or impossible to produce by hand. A CNC mill can turn a moderately creative person into an artisan wood carver in a matter of hours. Technologies that were reserved for extremely well-funded projects are now widely used.

The wide availability of digital manufacturing tools is allowing us to explore, for the first time, the world of 3D printing as a global community and share our experiences and designs with the rest of the world. 3D printing is

being used in almost every job field imaginable, from culinary masterpieces 3D-printed in chocolate to 3D-printed prosthetics and custom-fitted transplants for medical patients, these tools are changing the way the world itself is prototyped and designed.

For individuals without the capital to buy these tools, maker spaces are popping up in droves all over the world to provide makers with the tools they need, as well as provide an environment of collaboration and information sharing. Fablab, Hackerspace, Techshop, Makerspace—whatever you call it, there is likely one nearby filled full of amazing individuals building an endless list of creative projects. Many of these spaces are run as nonprofit organizations and operate simply for the creative good of their local community.

The maker community is not only changing the way products are designed and developed, it is changing the way products are sold. With the rise in popularity of crowdfunding via Kickstarter, projects that may have been considered risky to bring to market are finding their way into people's homes. Crowdfunding is a way for makers to ask their existing community to support their projects or causes financially. This removes the risk of having to predict future demand of the product before investing in materials and manufacturing.

In addition, social commerce sites like Etsy and Tindie make it possible to sell custom-made, short-run products to niche markets. This provides makers with a way to engage global communities for specialty products.

We are living in a time akin to the birth of the personal computer era during the early days of the homebrew computer club. One day in the near future, we will look back on this time as a pivotal moment in technology history when the way products are designed and developed changed and the objects around us became smarter and more dynamic.

This book will introduce you to people at the forefront of this movement, who are inventing these new technologies, building things, sharing processes, and changing the way we think and interact with the physical world.

Most projects start out as a need for the maker to scratch an itch or to materialize the vision of an object they wish to exist in their world. Sharing their creation and the story behind it is where the project ends and the maker movement begins. For me, the itch was simply a desire to read the stories contained in these pages. The object I wished to exist was this book. Hopefully, it will serve in some small way as my contribution to the maker community.

CHAPTER

1

Erik Kettenburg

Founder

Digistump

*As a boy, **Erik Kettenburg** taught himself electronics and programming. His natural curiosity for hardware electronics has led him to build some impressive projects, even at a young age.*

His recent Kickstarter project, the Digispark, was a big hit for providing a cheap, easily embeddable Arduino-compatible board for hackers and hobbyists. What started out as a way to scratch a personal itch turned into a wildly successful project. Erik continues to contribute to the open-hardware community through his website and online store at `Digistump.com`*, as well as exploring small-scale manufacturing methods, and performing contract work.*

Steven Osborn: It's always good to hear from folks doing interesting things here in the Northwest. I've only been here for a few years, but the maker community is really strong and growing. Anyway, tell me a bit about your background prior to Digistump.

Erik Kettenburg: Currently, my day job is the CTO of a company in Portland called Vacasa Rentals, which is a vacation rental management startup. I got there through being a web developer. Writing software has been my day job career basically my whole life. I was one of those kids who played with computers. When I was a kid, my dad introduced me to my first one, probably when I was four, which would have been an Apple IIe, maybe. I started writing PHP code when I was in junior high. And when I got to college—well, I had some small-paying gigs before that, got to college and took on a gig with a development shop, and I just worked my way up from there. This is actually

the first career job I've had where I've played with hardware at all, and that's been pretty limited to virtual phone systems and that kind of stuff.

Osborn: So then, what got you plugged into the maker culture and the hardware side of thing? What got you excited about making hardware?

Kettenburg: Hardware's always been my hobby in both electronics and computers. And I've always enjoyed that more but never found a career in that and didn't want to be a PC assembler or something. And that goes back to being a kid too.

When I was in high school, I built my first Tesla coil. I was always playing with motors and burning things and taking things apart. So I got involved first with microcontrollers, probably when I was in high school playing with PICs, back when you used a serial programmer for them or made your own for the printer parallel port. So that's always been a strong interest. And then, of course, Arduinos really revived that interest when they came out. I was maybe in late high school or early college when I got my first Arduino, one of the very first ones. And that was awesome because I didn't have all the time I used to have at that point, so having that pre-made for me was great. And then I pretty much just played with Arduinos in my spare time, and tackled projects here and there.

I built an automated chicken incubator, temperature loggers, that kind of stuff. And then I got into designing my own boards. One thing that always bugged me, no matter what my income was—it seemed like a waste of money to stick Arduinos in things. I always had like three Arduinos and twenty projects that needed Arduinos. So I was always ripping something apart.

I came across Kickstarter around about a year and a half ago, and found that there were some Arduino clone projects on there, but they all were very hard to use. Things I could use, no problem, but I didn't necessarily want to. I didn't want to build them or use a separate USB programmer and I didn't want to hand-solder a giant through-hole board and stick it in my project. So out of that, I got the idea of, "How small and simple can you make an Arduino and still have a plug-in that is Arduino compatible?" And so that started the Digispark. I had the idea, "I'll make a couple of these," and then the idea of, "Oh, maybe I'll put this on Kickstarter. Maybe I can make five hundred of them, and that way I can pay for the ones that I make for myself." And that's how the project started. That's why the initial goal was $5,000—because that was to make five hundred of them.

Osborn: So how many of them did you end up making? Are you still shipping them?

Kettenburg: To date we have made forty thousand of them. And we've sold about twenty-seven thousand of them, as of today.

Osborn: I want to backpedal a little bit and talk about some things you mentioned. It sounds like from a very early age you were working on—well, you mentioned a Tesla coil. That's a pretty big early project. Can you remember any things from back then that inspired you? Any projects you remember or people who inspired you to go out and build a Tesla coil or some of these projects that you've built? What was your inspiration when doing these things?

Kettenburg: I think a lot of my electronics interest just kind of evolved on its own, with my dad planting the seeds in that he had a big box of old electronics. The thing that I distinctly—not only distinctly remember but still have with me—is he had two Forrest Mims' Radio Shack books, which were *Getting Started in Electronics*[1] and *Engineers Notebook II*.[2] I don't know if you've ever seen the Forrest Mims' books, but in the seventies, I think probably when he bought them, they were *the* books—sort of an *introduction to electronics,* you know, a consumer book—and they were handwritten on graph paper. The whole book just looked like photocopies of somebody's handwriting on graph paper. I still use those books. They're a little outdated now in that they do everything with 555 timers and transistors, but that's probably the first reference I really read outside of talking to my dad about electronics. He's a high school teacher, by the way.

Osborn: You had this mentor in your dad that could help you get started and plant that seed, but from there you were self-taught?

Kettenburg: Definitely, definitely, yeah. I think he was very interested in electronics at one point, but he didn't keep up with it. It wasn't his hobby at that point. So it was kind of like, "Here's my old stuff. Here's how to do a couple of things." And I kind of took it from there. Of course, I didn't have access to the Internet for the first half of my childhood probably, and so it was pretty much limited to those books and playing with stuff at Radio Shack.

I grew up in Santa Rosa, California, which is about an hour north of San Francisco. We had a Radio Shack about two blocks from our house, so I would just buy things there and hook them up, see what happened. I'd often burn them out and spent all my allowance on that.

When we got the Internet, I was probably in junior high, and at that point, I got on the Tesla coil mailing list, and that's where that interest came from. I don't know why I got on there. I think that I saw my first Tesla coil in one of those catalogs they used to send out that were like called "amazing scientific gadgets" or something. I haven't seen one of those in a while. But that's probably where I first saw a Tesla coil, and I was saving up to buy one of the little $50 ones in the catalog that shot quarter-inch sparks, and I found my way onto the mailing lists.

---

[1]Getting Started in Electronics (Radio Shack, 1983)
[2]Engineer's Notebook II (Radio Shack, 1982)

I actually made some web stores for some of the people who were supplying Tesla coils, homemade Tesla coil parts at the time. And so then stuff started showing up on the doorstep, and my mom started asking why I was buying a four-inch Tesla coil. And I told her, "No, I'm building a 1.5-million volt coil." And there were some trips to Home Depot and some big transformers and stuff. I only ended up running it twice because I could tell it just scared the heck out of my mom, but my parents supported me in it. They even allowed me to drive a twelve-foot grounding rod into their back lawn so that I could run the thing. And that ended up going to Sonoma State University in Santa Rosa, where they were going to use it as a teaching coil.

Osborn: Wow. How old were you around that time?

Kettenburg: Let's see. I was in ninth grade when I built the coil, so what was I? Like fourteen?

Osborn: Nice. I'm pretty sure I just played *Final Fantasy* when I was fourteen.

Kettenburg: About fourteen, yeah. And I think toward the end of high school, when Arduinos came out—I don't know when I switched from PICs to Arduinos, but I got more into microcontrollers or more into small stuff that wouldn't kill me or scare my mom. And the biggest kind of lightbulb moment—starting with PICs and Arduinos—was the ability to interact with the computer, because at that point, I was saving for college by doing web development, doing a lot of contract work in high school on web development. So bridging that gap between hardware and software was really a lightbulb moment for me because I loved the hardware and I knew the software. That's remained my biggest interest, and that's why the Digisparks had to have a USB connector to it. I didn't want just something I could program and stick in a box to do something. I wanted an easy way to interact with computers.

Most of my personal projects with the Digispark have been using its ability to emulate a keyboard to make a random password generator or using it as a USB device to notify me over the network when my laser cutter's done cutting—pretty much simple tasks that would be expensive to do with a full-blown Arduino. And that's even more recently changed my way of thinking. Having thousands and thousands of Digisparks sitting around and knowing what their cost to produce is, it's an easy choice to say, "Oh! I could put a Digispark there." I think that's probably why it's been successful too.

Osborn: Say somebody who's new to the whole maker thing, but they want to get plugged into the culture or movement, or whatever you want to call it. Do you have any advice for the complete novice who wants to start learning things?

Kettenburg: With the Digispark, I see a lot of people who are just starting because I think the price point appeals to people wanting to jump in, but not willing to fork out one hundred bucks in parts.

I think my biggest piece of advice is to find something that you think you'll be passionate about, that resonates with you. I mean, the maker culture is so big now. There's 3D printing. There's electronics. There's all the subparts of electronics. There's the craft side of it, the clothes making, and then of course combining electronics and clothes. I just see so many different aspects of it. I see a lot of people jump into whatever's hot. "I'll get a Digispark because I want to be a maker!" But I think you need to pick an end result that would be cool. Are you after blinking LEDs? Or would you rather make a toy elephant, plastic elephant, or something? So I think that's the first thing. It's so big now that you can really just look around and pick where you want to jump in.

Then once you do jump in, I always jump into things with two feet for better or worse, so I always recommend that. Go for it and get involved in the community. I know we have a lot of beginners who read our forums[3] and don't ever ask questions. And they should be asking questions, because for the most part, the maker community is very friendly. More than any other community, there are so many people who are new to it. I would say that compared to Massimo [Banzi] over at Arduino or Bre [Pettis], you know, those people, I'm pretty new to it even, at least new to having it be a large portion of my life. And I'm still on those message boards asking the stupid questions and getting great answers. So I think the first step is to jump into the community. And there are so many communities to choose from. It's not hard to find one that both has the same interests as you do and is very friendly.

Osborn: So there's kind of the next step, a lot of people have projects or ideas that they're working on that would bring it to the market. Do you have any advice for those people?

Kettenburg: Build a prototype. I get so many e-mails about that. E-mails, Kickstarter messages, and they all start like, "I have this idea. I think it will be a great product. What was your secret? What should I do next?" All those kinds of questions. When the Kickstarter [initiative] was going, I'd get about one of those a day. Now I probably get about one of those a week. Mostly I get people who have an idea and haven't tried to build anything yet. I think often we get held up on, "I have this idea. How do I build that product?" And, really, the thought process needs to be, "I have this idea. How do I build—to use a startup term—how do I build the *minimum viable product*?" But a better question is, "What can I tape together, stick together to do what I want this product to do, even if it doesn't work right, even if it costs five times as much?"

Personally, I always get held up on things like, "I have to design the perfect circuit board." Instead, I should be building it on a breadboard first. So I think the best advice is to do it, build it, test it, and tell people about it. People are always afraid that their ideas are going to be stolen, but you know, you're

[3] http://digistump.com/board/

really not that important. Your idea isn't that important. If it's worth stealing, somebody will pay you for it then. And talk about it, be passionate about it, and actually build it.

I think once you actually have a working prototype, the way the economics of starting a hardware company or launching a hardware product, just now with Kickstarter, with all the clones of Kickstarter, is once you have a prototype, you can test your market really fast, which is amazing, because you don't have to invest the tons of money that I'm sure Arduino invested before they launched their product, and then all the money they invested to get their product out there and get it known, and how many years that took. Now you can do it in thirty days. I think it's important to not spend too much time building that first prototype, to not make it perfect, because the laws of business still apply: most things will fail.

I actually had a Kickstarter project before the Digispark that failed. It fell flat. Nobody was interested in it. And it was a software project. It was a software as a service dashboarding application, intended to provide real-time metrics. And I thought it was such a great idea. I thought it would be a big hit, and it fell completely flat. With the Digispark, I thought maybe I could sell one hundred of them, and it took off. So I think you've got to build it and test it, and test the waters and not spend too much time on the details until you do that.

Osborn: Great advice. Can you tell me a little bit about—can you give me some examples of hiccups you've had along the way or maybe speed bumps—unexpected challenges that you've had?

Kettenburg: Yeah, definitely. Well, the first unexpected challenge was we set out to raise $5,000—and we raised $330,000 and about another $60,000 in preorders.

Osborn: It's not the worst problem in the world to have.

Kettenburg: Yeah, not the worst problem, but logistically that's a big problem. I went from being a small customer for a surface-mount assembly shop to requiring a significant percentage of the factory, of the whole factory, to produce our product. As a point of reference, to produce thirty thousand of them kept the factory—a good regular-sized production facility—working for three weeks. I'm sure they had other little projects, but that was their main project for three weeks, the first thirty thousand. So that created a lot of scheduling problems, a lot of sourcing problems.

The first thing I found was there weren't thirty thousand ATtiny85 chips, which is the main IC on the Digispark. There weren't thirty thousand of them produced. So we had to go to Atmel, the manufacturer, and say, "You've never heard of us, but how fast can you produce thirty thousand?" This chip generally doesn't sell that well. I think ATtinies—used for hobbies, and another Kickstarter project, the Blink1—was also trying for the same chip and had

just been fairly successful. But, generally, the ATtinies were used in things like alarm clocks, where the production was scheduled months and years in advance, and they worked directly with the manufacturer. So we ended up with a ten-week lead time to get those parts. Ten weeks was the longest lead time for several parts.

Almost every part had a lead time. Most had to go back into production to meet our numbers. That took several weeks to get straightened out and get the right manufacturer who could make enough of them.

We knew—at that point, we already had this looming problem of shipping everything. We sold twenty-five thousand Digisparks through Kickstarter and the preorders that followed immediately. At that same time, we sold twelve thousand kits because we offered about twelve different shields,[4] you know, same idea as Arduino shields, but to plug into the Digispark. We did that because people were asking for it, not fully comprehending what we were getting ourselves into. By the time we shipped everything, we had twenty-six different things that could go in any given package, because we offered them both as a kitted version and as just the PCB board.

On Kickstarter, you're always getting pushed to set goals, bonuses when you reach a certain amount—stretch goals, that's the word I'm looking for. If the project reaches certain funding milestones you might offer add-ons or bonuses. So we did free stickers with that. We discounted one of our kits down to $1, which resulted in eight thousand of them being ordered. We had this huge logistics problem. We had to order all those parts. We had to kit them all into little bags, and we had to ship them all.

The biggest issue there was that the Digispark was so cheap that it didn't make sense to pay a fulfillment house to pack them for us. To have a fulfillment house pack them was going to cost about $6 on average per package—not including shipping—and we had put $5 of shipping into our price. So we ended up doing all of that in-house, my wife, Jenni, and I. We pretty much immediately started kitting things and ordering parts. We filled an entire bedroom, probably floor to halfway up to the ceiling, with just boxes of parts and boards. So that was a huge challenge, and it took us just about two months to ship everything, and we ended up shipping sixty-five hundred packages. I had never shipped anything commercially before that.

The other bump in the road—we ran into several bumps in the road with the factory—was that we were programming the chip in a slightly uncommon way, and so their programming hardware wasn't working. So we sent them samples. I hand-assembled some samples and sent them, and then they

[4] A shield is an add-on circuit board that attaches to the main microcontroller to provide additional functionality or peripherals.

tried to match that exactly. I sent them five test jigs on which you could hold a Digispark down and it ran through all these self-tests. They couldn't get it to pass the test because it wasn't being programmed quite right at the factory. That threw another month and a half into back and forth—videos, phone calls, e-mails—just trying to get it right.

Eventually, they got the right combination of programming hardware, and it turned out that they had tried to cheap out and bought this slightly cheaper version of the ATtiny85 from Atmel. Luckily, they had only gotten the sample quantities at that point. That version didn't work because it couldn't run fast enough to run the USB end of things. So there were plenty of hurdles there.

Then aside from the hardware side of things, we were trying to do something in software that had only kind of been done. Making an ATtiny chip talk over USB isn't a standard function of that chip, so we were completely bit-banging the USB functions. The version that I showed on my Kickstarter video was the first time I had ever gotten those functions to work. They still had some hiccups in them. I think it would have been no problem, again, if we had had five hundred or so, or one thousand, or even five thousand. But so much time ended up being consumed just by managing everything that it left a whole lot less time for the software end of things.

So in the end, we got help with the bootloader that would actually do the USB functions—do the programming over USB. You know, I had some starting points, I had it mostly working, and then one of our supporters actually stepped in and helped put together the final bootloader that we ran with, and that's an open-source project. In return, we essentially sponsored that project. We kept it, but we don't own it, we don't have any rights to it, so we used it and we sponsored it to keep them going and inspire them to get it done in time for the factory to ship it. I think in the end that was a really great outcome, because in the end that got the community more involved.

The bootloader created this entirely separate project that I know other people are already looking at. Adafruit has a device called Gemma, a device coming out that's based on the ATtiny85, pretty similar to the Digispark. It's made to be sewable. I think it has three input/outputs instead of six, but I saw they're considering using that same bootloader. And people have started contributing to it. It's now on GitHub. There's now like three different versions of the bootloader. When you first power up a Digispark, there's a five-second delay before it runs your code, while it waits for a USB programming signal. Some people didn't want that, so they made the microcontroller check if a pin is tied to ground instead, and if it's not tied to ground, then it boots right into the user code, so already that's evolving. So that ended up being a bump in the road that turned into an even better thing, and a completely new project launched out of it too.

Osborn: That really is a nice result, saved you some time and really got the community involvement going. Building a community behind these projects is a great outcome.

Kettenburg: Yeah, definitely. And I've spent a lot of time on the community end of things like, "This library could use improvements," or "We could use another release of the software that fixes the known bugs so people don't have to download this file or that file depending on which version of Linux they're using." All those things are on my list, but an emphasis has been on the community and it's paying off. We have a fairly active set of forums, fairly active in posts, and extremely active in views. People are helping each other on them a lot already.

I would look at new topics on the forums and anything that hadn't been solved, I would try and help out with. That went from taking me hours a day to now probably fifteen minutes a day because we have some real dedicated community members helping out, and then a lot who just help out here and there when they see something. That's been great to see. I think one of the challenges for Digistump as a company, with the Digispark and with other products I have in the pipeline now, is I'd like to see the Digispark community and the Digistump community tie into the broader Arduino community more. You know, you have people running official Arduino hardware on their Arduino forums. You get people running the Teensy[5] hardware by Paul, who is in—do you know Paul? I can't ever pronounce his last name.

Osborn: Stoffregen, yeah. I've met Paul a couple of times. Seems like a nice guy and his product is super useful.

Kettenburg: He's right outside Portland there, and he's got a very active community. He's great with his community. He offers a lot of really technical help. He's pretty undisputed in his technical expertise on all things microcontroller. And we have all kinds of distinct communities that grew out of the original Arduino community, and they don't tie together very well. Adafruit is going to have their ATtiny-based device and Todd Kurt—the maker of the blink(1), BlinkM series of devices, down in LA—has a community, and the Digispark has a community, and we're all running the same hardware. So I think that's maybe a challenge of the maker community at this point, or I see it as one, that we need to join together a little more and tie that information together a little better. And the same for Arduino libraries. The other day, I was looking at the universal TFT library for driving color LCD displays, and it's like there's this really nice universal library, and then Adafruit has this really nice library too. I can't imagine how much time was spent on both of those

[5] www.pjrc.com/teensy

libraries, and then in the end, they do the exact same thing. They both do it almost the exact same way. They both have the same issues remaining that no one seems to have time to get to.

Osborn: One thing that I've seen that kind of bugs me is not only that there's duplication, but finding the right library for what you're doing is somewhat difficult. And then for a lot of people, just figuring out where to put those files in the right place in the file system is a challenge. If you're new to it and you find this library, what do you do with it? So I've always thought it would be a great project to see somebody build a package manager.[6]

Kettenburg: I was just about to say that, yeah! With the Digispark, we have to modify the Arduino distribution slightly, and it's such a pain because it's not built for that. The Arduino guys, all the thanks to them, created this whole Arduino ecosystem. They started it all. But they're relatively resistant to compatible hardware being made truly compatible. And I think that's seen most in that there's no plug-in system for the IDE to support other programmers. There's some there. There are board files and you can define other devices, but it still assumes that they're exactly like the Arduino. So you got Paul, who had to make a special version for his Teensy, and we had to make a special version for the Digispark, and it seems like a package manager could solve all that too.

I've put a lot of thought into how to make a good package manager for it, but it's kind of one of those jobs that take a ton of work and very little payoff. If I had, if the Digispark hadn't happened, that's the kind of thing I would pursue but don't have the time now. But it's certainly something that the community needs. I've even thought about making a Kickstarter project for sponsoring a package manager, and then hiring someone to do it, or something like that, because Digispark/Digistump would benefit from it greatly.

Osborn: This is something the community needs.

Kettenburg: The next project I have in the pipeline is probably going to need modifications to the Arduino IDE too.

Osborn: Yeah, I definitely think that's something the community needs. I've looked at the internals too. Even in places where they had made some effort to make things more generic, for instance, the Avrdude uploader. There's an uploader base class and then there's the Avrdude subclass, but the base class has all this very specific code that makes it not generic at all. Even the places where there was some effort or forethought to make it generic, it has been bastardized to the point where it really no longer functions generically.

[6] A package manager is tool that manages project code requirements by automating the downloading, discovery, and updating of libraries the project is dependent on.

It's certainly not insurmountable by any means to accomplish this. It's going to take some work cleaning it up, I think, before you could start really building on top of it.

Kettenburg: Yeah, definitely. The biggest incompatibility we had integrating the Digispark into the ID was that you have to use Avrdude, and Avrdude didn't work with our bootloader. We needed a specific specialized upload tool, and we had that in a command-line form, but I ended up having to replace Avrdude with a dummy executable that's called Avrdude, and then rename the real Avrdude, and then pass the flags over to that unless the flags say it's Digispark, and then we pass it to our command-line program instead.

Osborn: Wow.

Kettenburg: It's one of those I could have gotten as an Avrdude source and added support for the Digispark, but you know, then it's like we're getting into branching yet another project, and in the end, it seemed cleaner just to put a proxy in between. And, the—oh, I can't remember the project—one of the robot projects off of Kickstarter, they were taking the same approach, and we exchanged notes to make sure that our proxies wouldn't conflict with each other.

Osborn: That's pretty clever. It would be great to see the community take a bit more forward-reaching approach there, a bit broader approach.

I was wondering what new projects do you see out there that you think are the most interesting? What other things have you just looked at recently and said "Man, that's cool!" A project that someone's building?

Kettenburg: I think that some of the coolest ones to me—and I guess my view, my interests, because of the Digispark and because of trying to build a company off of that, have kind of turned to an interest in do-it-yourself manufacturing, the makers who are manufacturing themselves and building their manufacturing equipment. We're moving that way. Any day now, our first pick-and-place machine will be delivered. It's this little desktop unit that they build in China, and it's pretty much not exported at all. And these Chinese makers are using it to make one hundred boards at a time in their living room.

Osborn: I'd love to see that sometime.

Kettenburg: You know Zach Hoeken, the former cofounder of MakerBot? He lives in Shenzhen now. And has a blog called Hoektronics.[7] And he has a video of the machine that we bought in the form of an interview with the maker in China, who uses it to make boards in his living room. That's where I first saw the machine.

[7] www.hoektronics.com

Osborn: I think I might have seen that video. I'll have to take another look.

Kettenburg: Ian over at Dangerous Prototypes—are you aware of Dangerous Prototypes?[8]

Osborn: Yeah, I've got a Bus Pirate.

Kettenburg: So Ian was over in China and saw these too and ended up ordering one for himself. Then he organized a group buy with about seven of us who all bought them direct from the factory and ended up getting a big discount. We ended up paying less than they sell for on the Chinese market actually, so that was great. The manufacturer was so excited to be exporting them to America, because there really hasn't been any market for small pick-and-place machines over here.

So that whole movement of really small-scale manufacturing in the US is something that I get really excited about. And that's not just premade things, like the pick-and-place machines from China, but people who are developing homemade laser cutters. The 3D printer movement is awesome, but I still don't see 3D printers—you know, a homemade 3D printer still isn't practical when I need to make twelve hundred cases for Digisparks that I've sold. So the laser cutter still wins there. I have two laser cutters, and one of them is a big, giant one that we air-freighted from China. It's a beast. It's a great machine, but I've seen a lot of these homemade laser systems—on BuildLog and other sites—coming up, and I think that's really cool.

Osborn: I bought a Full Spectrum 40-watt hobby laser recently. Haven't made anything exciting with it just yet.

Kettenburg: I have that one. Do you have the fifth gen or the fourth gen?

Osborn: I have the fourth gen.

Kettenburg: Okay. I've got the fifth gen, and I liked it. I've kind of soured on Full Spectrum since then, since I've seen how they've treated their customers. I liked the machine. It couldn't keep up with our production. We bought it thinking we're going to make, like, one hundred acrylic cases, and we ended up with orders for twelve hundred, so that's why we flew in this beast of a 60-watt cutter from China. That machine is actually a beautifully made machine, despite what everybody said about buying a laser cutter from China. Of course, we researched the company and everything, and it's built like a tank. It cuts about six to ten times faster than the Hobby one, and so that's amazing to watch.

I need a reflow oven to go with the pick-and-place machine, and I don't like the Chinese options there. They have a lot of safety issues, a lot of fire issues, and generally just don't have the reliability that I need. And we're a little too

[8] http://dangerousprototypes.com

big for the hot-plate method. I'm going to be doing twenty-inch by fifteen-inch panels with, say, one hundred to two hundred boards on one panel. So I finally settled on it last night. I'm just going to build a reflow oven from scratch. I see other people frustrated with the choices, so I wonder if that's an area that's maybe about to kind of kick off in the maker culture, because people are getting into this lower-quantity production.

I know a lot of people do the toaster ovens, but you just can't get a big enough toaster oven once you're making one hundred boards at a time. So those projects have really excited me.

On the embedded microcontroller side, there's actually a project on Kickstarter right now that's an FPGA.[9] I think that's really cool because I've always wanted to work with FPGAs. And really, there isn't any kind of mainstream, real accessible development board out there for it. The project's called Mojo.

Then there's Parallella—I'm not sure if you've seen that project. It's a parallel computing project. That was probably about four months ago or so. I ended up backing that, and I'm really excited for that kind of stuff. I guess in general, this is like more advanced computing. We have the 16 megahertz Arduinos down. Now it's time for the more advanced but still very accessible options, like an FPGA, like a parallel computing board.

And the next Digistump product that I'm working on is actually a 32-bit platform similar to the Arduino Due, similar in that that's the closest thing out there to it but with embedded Wi-Fi on it. I think that's the next step—making wireless connectivity ubiquitous on these devices. And right now, it's not only really expensive, but there are just so many wireless standards out there. Putting Wi-Fi on your board is resource-intensive and expensive, so then you put an XBee on, but then you need a board to receive the XBee signal. I think the real solution is just to get a Wi-Fi-enabled board down to a price range where every device you make can just have Wi-Fi on it. Or another protocol. It doesn't have to be Wi-Fi, but really bringing those prices down is necessary. My target price for this project, not completely confirmed, but kind of the target price I'm designing around is to have a Wi-Fi-enabled, 32-bit, Arduino-compatible board. Right now we've got about one hundred I/O lines on it, and we're looking at a $35 to $45 price range.

Osborn: Yeah, that would be pretty massive, I think, for the community. People would fall all over that.

Kettenburg: We're hoping so.

Osborn: I would buy a couple. I'm just saying.

---

[9] An FPGA, field-programmable gate array, is a hardware device designed to be configured by the end-user after manufacturing.

Kettenburg: Once you put even a cheap Wi-Fi board on a Due, an Arduino Due, you're at like what—$110?

Osborn: Yeah.

Kettenburg: I'm not going to spend $110 on a project even now.

Osborn: Yeah, with the cheaper $60 WiFly shield and a $30 Arduino, you're still at looking at quite a bit of cash.

Kettenburg: Yeah, you're still at ninety bucks. So I mean, it's not going to be an easy one to pull off, but the Digispark was a lot easier from a hardware standpoint. This one should be more straightforward, or is more straightforward from a software standpoint because we're not trying to do anything with hardware it's not made to do. You know, we're not bit-banging USB or anything. The challenge with this one is purely hitting production numbers to be able to source those parts at those prices. I've been working with some different companies in China that produce Wi-Fi modules to get them in the right price range. We haven't decided exactly how we're going to pursue the funding, but it very likely will be Kickstarter.

And for this one, I'm going to do kind of a prefunding round through our web site with our existing backers and say, "Do you want to be part of this? Do you want to get one of the first prototypes? For one hundred bucks, you get one of the first prototypes and you get one of the final products when it's done." And use that money to essentially go the factory and say, "We want to make one hundred of these, but you can't hand-assemble them. You have to run them like they're a full production run," so that we can work out the kinks and everything before we set a final price. We would like to make sure that we keep the margins tight because we truly believe in making it accessible, but I would like to do this full-time someday, and so it would be nice if we made a little more off it than we made off the Digispark.

Osborn: I just have one more question about Kickstarter. I mean, everybody and their dog is launching Kickstarter projects it seems, and I just want to know about your experiences with it and maybe any hiccups you had or challenges dealing with the Kickstarter model. It sounds like you're thinking about doing it again already, so I'm guessing you had a pretty good experience. Is there anything you can say there or any advice you have for anybody considering using Kickstarter?

Kettenburg: Yes. So we had a couple of bumps in the road with Kickstarter directly, actually. Kickstarter pulled our project when it was around the $200,000 mark. It still worked as a project page, and it was still taking pledges, but you couldn't find it from any of the Kickstarter pages. You could only come in from a direct link. I guess that's what they do when they're pending a resolution because they don't like something you've done. And the communication

there was definitely a lacking. We were getting hundreds of messages a day through Kickstarter, and their message came like any other Kickstarter message, and I missed it for a whole twenty-four hours.

So the problem was we had packages that had too many units in them, and we had mentioned that you could ask for them to be in retail packaging if you bought over two hundred of them, and Kickstarter didn't like that. It admittedly wasn't consistent with their guidelines. But I didn't think of it at the time we offered it, because somebody asked, "Can you do this?" And essentially we had to get those people to cancel their pledges, because when you're running a Kickstarter campaign, you can't cancel someone's pledge on your own. "Get them to cancel their pledges, remove that category, or we'll take your project down." And the only way to communicate back with them was through Kickstarter messages, you know, through their ticket system.

With such a huge project and so much at stake, it was a frustrating couple of days. It took four days for them to put our project back on all the pages. So it definitely was difficult to communicate with them. That was a challenge. We felt when a project reaches a certain size, and with the problems it can encounter, there should be more support there. If nothing else, a number we could call and talk to someone at Kickstarter to get things resolved. I've heard that same sentiment echoed by other big projects.

Eric, the guy behind the Pebble watch, is a Digispark backer, and he and I talked actually quite a bit about issues with Kickstarter. He tipped me off to another potential hurdle with Kickstarter. We knew there was this hurdle that you had to wait fourteen days for Amazon to release your money after the project ended, which in our case would have meant waiting fourteen days to authorize production, because I didn't have enough money in my bank account to fork over $150,000 for production. So he tipped me off that you could just ask Amazon to waive the fourteen days. Essentially, I sent them an e-mail and said, "I've been in business. I've had my own consulting business for *x* number of years. My credit is fine, and all my tax information is entered. Will you waive the fourteen days?" And they said, "Sure, no problem," and they released all the money to us two days after it funded. That certainly was a nice piece of advice. Amazon Payments is actually really easy and good to work with, and supportive of Kickstarter projects. They realize there are some unique challenges there.

The other challenge with the platform is that Kickstarter still isn't built for the volume of business that's taking place on it. You have Kickstarter messages you can send to your backers, and you have Excel reports that you can click a button to generate, wait for them to generate, and then wait for them to download. And there's one report for each reward tier. There's no API. There's no master data dump. There's no way to send automated messages. So then we screen-scraped everything. I wrote screen scrapers, and we were using cURL to fake our sessions and our logins to automatically notify backers

of different things to download the reports and update our databases. So not only did we have to build an entire system on our own to handle the sales and backing, keeping track of pledges and all that, but there was no way to directly interact with their system. We had to do all this in roundabout ways, which certainly led to some pretty major hiccups, and if nothing else seemed like a waste of time. But in the end, we couldn't have done it without it. I get a lot of e-mails from new Kickstarter-type sites saying, "Launch your project with us. We have fewer fees. We have this, we have that," but the visibility of Kickstarter—you can't beat that.

Overall, I'd never be one to complain about anything that enables me to start a business with essentially zero capital. And I actually have a degree in economics. I studied business. I've worked with lots of startups. I've seen how much capital people can burn through, and I understand investing money to make money and all that, but I never dreamed I could start a business with essentially none of my money at risk. So overall, I would say it's a great experience.

There are certainly some big challenges there. I guess I'd add that another challenge at Kickstarter is that there's no limit to the funding. You can limit your rewards, but no one is going to limit their rewards. So I think a lot of the projects I see having a lot of issues result from ideas that aren't fully fleshed out and then run a while. And a lot of times, the amount of money those ideas were asking for in the first place—if they had only got their $500, it probably would have come out fine. They probably would have made their backers happy. But then when they have $50,000 and still don't have a fully fleshed idea—and I don't know what the solution is there, because that's certainly part of Kickstarter's success too, but that can be a challenge. You can go to sleep and wake up and have a much bigger problem on your hands.

Osborn: There's definitely some nuance on Kickstarter's part to balance people's expectations with the project's risk. I mean, there's only so much that Kickstarter can do to make sure the project is going to be successful, but if it is not, people tend to be upset that they invested their money in it. Personally, I feel like I want to back some projects where there is some risk involved because those are the projects that are doing new things. The projects where there's no risk involved are just basically established companies that are releasing the new version of their product, and that's not really moving the needle forward, right? I mean, there's definitely a lot of balancing of priorities and risk, I think for everyone there.

Kettenburg: I think a big threat to Kickstarter from within is the number of commercial products. And when they changed their rules[10] within the hardware category—that you could only sell one unit—it would have killed the Digispark! Well, it wouldn't have killed it. We would have made our five

[10] www.kickstarter.com/blog/kickstarter-is-not-a-store

hundred. I would have been happy, but it wouldn't have been the success it was, because something like seventy percent of the backers were buying three or more, maybe even more like eighty-five percent. About half of all backers bought three of them, which I figured, if I had seen the project, that's probably what I would have bought, because it made the most sense to ship them at that price. So I cringed when they did that. And a couple of people interviewed me about that, and I talked about how I felt that that was a bad move. I still don't think it was a good move, but I do understand the problem they have there of wanting to support projects and not just the sale of finished products.

The other thing they did is they made the hardware category not just open source. I think that was the worst move because I think that has encouraged commercial products and I think it's also encouraged people who have more of the mindset of looking to make profit than looking to launch their projects. That's kind of how I think of things: there are products and there are projects. The projects can have a product, but when you look through Kickstarter, you can often clearly see who's trying to sell you a product and who's trying to launch their project, and that's kind of what influences my backing of things. I want to make sure that even though we have this little company now, that the next Digistump project that goes on Kickstarter is truly a project. It's doing new things, it's opening new doors, and it's providing something for the community.

An example of one I've felt a little torn about right now is the Hydra bench-top power supply. It's a cool idea—a lot of people in the electronics community have been really getting into these homemade bench power supplies that run over USB, and you can set them and log from the computer. And that was a realization of it, but I just can't get over the fact that it's not open source because I can look at their board and see that they've pulled from everything out there, all these projects from the EEV Blog,[11] from Ian's blog at Dangerous Prototypes,[12] you know, all this work people have been doing. And it's on Kickstarter and it's really killing it. I look at the comments, and I notice that no one's even asked them, "Why aren't you open sourcing this?" So, you know, coming from the Digispark, which wouldn't have been possible without all the open-source projects before it, I have trouble with that.

I see the commercial versus open-source issue that MakerBot is very publicly dealing with right now, and I don't necessarily agree with the route they've taken, but I certainly am carefully considering that for our next product. I've heard that DF Robot, one of the Chinese companies, is already working on a Digispark clone, so, you know, that issue is alive and well. But I do truly believe that if your product is successful, good, and unique, and you support the community, the community will support you.

[11] www.eevblog.com
[12] http://dangerousprototypes.com

I think that really shows in Arduino. There are so many clones in Arduino, and there are even mainstream clones and American-made clones of it, and still, most people buy an authentic Arduino. I own Arduino clones, from Seeed Studio and the like, but I still buy the Arduinos. I do see a place for licenses, like creative commons, noncommercial, and that's what I'm considering for some of our future products. Just to say, "This is for everyone. But if you want to make a product off it, you'd better make it a new product." And that's why we actually chose to make the Digispark completely open source. But the name is not open source. It's kind of like Arduino did the same thing. You can't make our product and call it a Digispark. And I get a lot of inquiries about, "Can you send me the files? I live in this country and I'm planning to produce the Digispark because you're selling it for too much." And I usually respond, "We really prefer that you license it from us," or "We'll give you wholesale pricing. We can work with you, but the files are public." And some have even gone so far to send follow-ups to tell them all the details of how it's produced and what they should tell the factory and everything.

Osborn: Oh man. Yeah, you don't get everything for free. You have to do some work if you're going to steal the whole item.

Kettenburg: Yeah, I tend to emphasize to those people that there's test jigs involved. There's all this stuff you're going to have to do if you want to produce it. We didn't release the plans for our test jigs. If some maker e-mailed me and said, "I'm trying to make a test jig for my product. I'd really like to see how you made yours," you know, I'd send it to him in a second, but I think there is something to be said for finding that balance as an open-source company. And you do have to protect it somewhat, but I'm afraid that people are starting to take that too far. And I think the public manifestation of that is MakerBot.

Osborn: It's interesting to see this evolving in the hardware space, because I've seen this kind of thing in software for a long time. I think WordPress is a great example of an open-source community. `Wordpress.com` powers a large chunk of the Internet, but they also enable all their competitors to go out and just roll their own WordPress and self-host it. There are pretty successful companies whose business model is hosting WordPress and they enable that.

Kettenburg: Yeah, definitely. Being a PHP developer, I've never been a WordPress user, but I've seen people make a lot more than me just installing WordPress. People seem to be making a very big deal about the hardware world facing these challenges, but it's only a big deal because the hardware world has been so locked down historically. Even if the MakerBots go and come up with some quasi-open-source business plan, there's still going to be the Adafruits and the SparkFuns who—well, SparkFun is kind of in between, because they use a noncommercial license too. And in the end, we're still miles ahead of where we were ten years ago in the hardware world.

Osborn: Yeah, and there's the open sourcing of the firmware and the software that's running on the device, but then there's the actual physical open source. Open-source schematics, open-source silicone, and stuff like that, which is—even on the projects that are very open, like Raspberry Pi, Broadcom has no interest in showing you how they make their processor.

Kettenburg: I have a lot of reservations about Raspberry Pi calling themselves open source, not just because of Broadcom, because they have kind of taken this attitude of, "Our hardware is open source, but you don't really want to see it because you'll never be able to make it anyway."

And then on top of that, they represent themselves as a nonprofit, but make far more units for sale than for nonprofit use. They're a really unique case of it because in the end, they released their schematic. I looked at it and saw a lot of ways they saved money and thought, "Well, this is a great guide." Someday, if I can do this as a full-time gig, I would love to make a true open-source Raspberry Pi, as many have already. So that's certainly some useful information to the community. And there's no denying what the Raspberry Pi's done for the ubiquitous and embedded computing community. But, yeah, they're a very interesting case. Very mixed feelings there.

Osborn: Well, all right. That's some great insight. Especially into the sort of challenges that come with Kickstarting a physical product and getting it through manufacturing. I'm looking forward to seeing what Digistump has in store for the future.

CHAPTER

2

David Merrill

Cofounder

Sifteo

David Merrill *holds a Ph.D. from the Massachusetts Institute of Technology (MIT), where he was part of the highly regarded Media Lab program. He has devoted his studies and career to exploring alternate user interfaces and interactions between humans and physical computing objects. He now runs a company called Sifteo (sifteo.com), which has commercialized the work he began at MIT.*

The first time I saw a Sifteo Cube, I recognized that it was something totally unique. They are simple to use and understand, yet challenge our assumptions and conventions around how a computing device can be used.

Steven Osborn: Tell me a little bit about the projects and the people who inspired you to make things.

David Merrill: I think the reason that I started making things came from my dad. He should have been an architect or a mechanical engineer, but he didn't wind up studying either of those disciplines, so he wasn't. But he was a really mechanically inclined guy and was really good at sketching and building. I have this one picture that he sketched for me after he had disassembled my bike to ship it to me on the East Coast. He had drawn an exploded-parts diagram, basically of the whole handlebar assembly with arrows and labels, so that I could put it back together once I got it. He was a natural maker.

When he was younger, he built things like musical instruments. He built a couple of mandolins. When he became a homeowner, all that energy was directed at improving the house. I saw him re-roof the house and build additions, and he once built me a lofted bed in the bedroom I shared with my brother. He would pour new concrete paths until they went all the way around the house.

So I grew up with this role model in my family and a feeling of being empowered. If you had an idea for something you wanted to build, you figured out how and did it. I think in addition to that more abstract feeling of empowerment, just watching him build and solve problems gave me an intuition about problem solving. That was where it all started for me.

When I was a kid, I used to build things too. I built a coaster car, a wooden car that you pushed up a hill and then rolled back down with gravity. It had a steering wheel and brakes that were basically a piece of wood on a pivot that you could pull up, and the other end would scrape the ground to slow the car down. What else? I built a minibike with a lawnmower engine, my own little DIY motorcycle from a kit, basically. I had to go find the engine and put that together.

Osborn: Did you scare your mother to death with that one?

Merrill: Yeah, yeah. I think the reason that she let me do it was that my dad was very supportive because he had actually built something just like it when he was thirteen years old. I think he loved the idea that I was following in his footsteps and wanted to build this little minibike.

I tried to build a miniature steam engine once out of wood and it turned out it wasn't airtight enough—none of the seals were sealed enough to actually make it work. But I loved physics class in high school because we had to build little pneumatic cars with a two-liter bottle and pump it up with air to see how far you could make your car go. We built a little roller coaster for marbles with weed-whacker cord as the tracks. So that was just the childhood I had, loving to make things. I had a lot of physical toys like blocks and Lincoln Logs. My earliest blocks were a set that my dad had made out of two by fours that he had cut down to a couple different lengths and standard sizes. He gave me a whole set. I grew up really thinking that one of the fun things you could do in life, or if you're bored or whatever, was to make things, or think up good ideas for making things. So that's kind of the background of my personality and why I love making things even today.

There are a couple of other key moments I would point to. When I was probably seven or eight, my dad got a computer for our house. That was when I started having access to a computer. I mostly just played games on it. I didn't start programming at home at a very early age. But when I was about nine years old, I got exposed to the Logo programming language when I went to a summer day camp. I didn't think about it that much after that summer, but I think it planted a seed for programming. I had Logo access in school, too, when I was in fourth grade, which continued that interest. I built up some pretty elaborate animated scenes, almost like I was directing a movie using Logo, by creating very simple graphics and moving them around programmatically. Then, I started playing videogames, although my parents never let me have a game system like a Nintendo or PlayStation, or anything like that. So I had to play shareware games on the computer.

I kept playing whatever videogames I could find, and then when I was in high school, my next programming experience came when I was taking calculus as a senior. It turned out that I could write programs in BASIC on my graphing calculator to make some tiny programs that were useful to the other students in the class. So I really got into that.

I think the thing that was so satisfying about that was that I got to make something that was useful for other people. Since then, I've recognized that that's one of my primary motivators in life, and in my pursuit of technology, and in entrepreneurship. I just love making things that are useful for other people, and then putting those things in their hands and getting the feedback that yes, indeed, it is actually a useful, life-improving thing. So that was, I think, a transformative moment in high school.

Osborn: What was the nature of the program?

Merrill: I think it was a successive approximation under the curve to do a discrete simulation of an integral. I wrote this program that would iterate and you could put in the bounds and what you wanted the step to be each time, and it would calculate the area under the curve with a bunch of rectangles. That was a satisfying thing because I made this program and it worked. I debugged it and made sure it worked for me, and then I handed it off. Everybody else in the class started using it on their calculators. So that was—before anyone was doing much with the Internet—my first experience with sharing software and experiencing that exponential, multiplicative growth of utility that comes from software being able to replicate. That was really cool and fun.

When I got to college, I thought I would be a physics and Spanish major. I loved both of those topics in high school. I think what I loved about physics was that it got me playing with things that were really more like mechanical engineering concepts. I learned pretty quickly in my freshman year that physics wasn't for me, after I got an A, then a B, and then a C—one quarter after another.

But what I discovered at the very end of my freshman year was programming. I took the introductory programming class—this was at Stanford, Computer Science 106A. It was a programming class that massive numbers of students on campus take. I think it's the most popular class on campus. Especially at that time, when the Internet was getting going. This was 1996 or '97. A lot of people wanted to know how to program. So I took this intro class and totally loved it. It brought back this experience from senior year, where I had made this program and shared it. The exciting thing about it for me was that there's a certain level of behavior and utility that you can make with physical stuff, but you can breathe a totally different kind of life into something when you can program it with code. And that was just so exciting.

I wanted to make videogames. That was my motivation at the time. I grew up not having a Nintendo but loving games like *Super Mario Bros.* And my primary motivation at the time was, "I need to get good enough at programming so

that I can make a game like that." And so I got into computer science and started taking more and more classes.

Osborn: So you ended up majoring in computer science?

Merrill: I did not end up majoring in it, but majored in an adjacent field called Symbolic Systems. At other schools, they would call it "cognitive science." It's computer science, psychology, linguistics, and philosophy all rolled into one. So I got a broad base of experience in computer science and really loved learning about perceptual psychology too. I had a couple of classes that really shaped my way of thinking about technology, because they were really about us humans - about the human perceptual system and body and how that related to technology.

One was about the human vision system, like how do our eyes work? The first half of the class was about that, and the second half of the class was, "Okay, so now we know how people's eyes work. What does that mean for how we should design imaging technologies like printers, and displays, and projectors, and things like that?" I thought that was super cool because it was a way to ground our design of technology in the truth of how our bodies and brains work. That's something that changes much more slowly than technology. We're not evolving anymore, but we can change the way the technology works to match us.

So, anyway, to wrap up this part of the story, I took a bunch of computer science classes. And that was great and fun, and I kind of expected that I would leave school and go into doing something related to software. But then I started to take computer music classes at the Center for Computer Research in Music and Acoustics (CCRMA) at Stanford, and really got into that combination of music and technology because I'm also a guitar player. And one of the key things that happened was I took a class taught by Bill Verplank that was about designing and building new music controllers—new musical instruments out of technology.

The key thing that happened in that class is I realized what I could build with hardware, too, because the class taught some very basic electronic circuits and simple microcontroller programming. At the time, we were doing BASIC Stamp programming. We put together our own custom controllers with buttons, and sliders, and force sensors. I used cameras. It was this amazing revelation for me! My palate of tools expanded and I could build interactive circuits that could explode beyond what was inside the screen, and keyboard, and mouse of the computer. I didn't have to make things that just lived on the screen in the software of my laptop computer anymore. I think the reason that was so exciting and so transformative for me was that I grew up playing and building with very physical things, like blocks, and building things with wood and metal. All of a sudden, my two worlds of building stuff, one from childhood, one from college, had found a way to meet each other. And at the