TECH REIMAGINED
AN ENDAVA PODCAST

CELEBRATING PROGRAMMERS' DAY AND ALL THINGS TECH

[00:00:11]
BRADLEY HOWARD (BH): Hello everyone. I'm Bradley Howard and welcome to the latest episode of Tech Reimagined. Today marks a special day in the history of technology. It's Programmers' Day. To celebrate, we've dedicated today's episode to the special occasion and invited a dear friend of ours, Guy Kawasaki, to talk all things tech. Hi Guy, how are you?

[00:00:30]
GUY KAWASAKI, CHIEF EVANGELIST - CANVA (GK): I'm good, thank you. Thank you for reminding me. Is this like an official holiday?

[00:00:35]
BH: It's the 256th day of the year. Two five six four two to the power of eight.

[00:00:42]
GK: Oh, is that how they got it really. I've been missing - I learned something new today. Alright.

[00:00:55]
BH: So Guy, back in the days when you worked at Apple, you were the Chief Evangelist and you were essentially marketing to programmers to create more applications for the Apple platform. How did that go? How did you find the programmers? How did you interact with them? Because I don't think you were a programmer before Apple.

[00:01:11]
GK: No, no, no, no. So I'm living proof that you can fool most of the people most of the time. I have a degree in psychology, which is the easiest major I could find. And then I went to law school for two weeks and quit. Then I went into the jewellery business and then I went to a software company for a very brief time, like less than six months. And then I went to Apple to become Apple's second Software Evangelist, convincing people to write Mac software. So in a nutshell, that's my checkered past. So I've never taken a computer programing class. And yet I had to deal with programmers because it was my job to convince them to write Macintosh software. And back then to tell you how crude it was, almost embarrassed to tell you, you had to buy a – Lisa computer. And a Lisa was, I don't know, seven or ten thousand dollars, so you program on the Lisa and then you compiled for a Macintosh and so you had this cross compilation system.

So you weren't even programing on a Macintosh for the Macintosh, you were programming on a Lisa for the Macintosh. And we had this documentation called Inside Macintosh, and it was printed on paper of the quality of a telephone book. So, you know, not exactly the thickness of best paper. And it was thousands of pages. And so I would meet with developers and there are kind of three pitches when you met with a company. So pitch number one was sort of the engineering and tech, which is we have so many calls to the raw and we do so many things that you can finally write the software you've always dreamed about. So that was the weenie pitch, right. There was also the pitch like, oh, you know, computers are so hard to use by having a Graphical User Interface. We can introduce a computer that more people can use. We're going to broaden the market for personal computers. So that was the marketing pitch. If you met with the marketing people, the company, and then if the finance people are, you know, the adult supervision was in the room, then we would talk about, well, you know, right now you're writing IBM PC software and IBM is publishing their own software. They have their own software publishing division. So imagine if they created a title and bundled it with their PC, you'd be out of business. So, you know, you can't be on only one horse. You need to be on at least two horses so that if one horse throws you off, you

can go to the second horse. So we gave these three pitches and inevitably company self-selected. I mean, you could tell that - oh, yeah, I could finally write the software I always wanted to write or oh yeah, we're going to we're going to increase the market for computers or, oh shit, I don't want to be totally dependent on IBM. And one of the lessons I learned of evangelism is that if someone is open to being evangelised by you, they will tell you how to sell to them. And all you have to do is be smart enough to be quiet and listen and regurgitate back to them. What you heard them say, and then they're going to think you're so smart. And so perceptive because you basically are saying the same thing to them, they said to you.

[00:04:39]
BH: So who was it easier pitching to the developers, the marketing team or the finance team?

[00:04:44]
GK: This is usually the developers because the developers saw Macintosh as such, the holy grail of, oh, my God, you know, look at this display, look at the WYSIWYG, look at the ROM calls. You know, there's, I don't know, 400 calls to the ROM. And, you know, we don't have to be screwing around with drawing rectangles and, you know, all this kind of stuff. And so for them, it was, you know, hallelujah. I mean, it was Christmas every day. It was Programmers' Day every day when Guy showed up with his Macintosh prototype,

[00:05:21]
BH: it was Programmers' Day every day. Oh, I love that. So how has the role of programmers changed now that when you were originally pitching the latest Apple products, they were getting pretty low level, like you say. I mean, you mentioned ROM, not us. Yeah. Now that's been extrapolated quite a bit and they're working with all different types of automation software. What do you think the role of the program has changed?

[00:05:49]
GK: Well, I have hardly a random nor representative view of the world here living in Silicon Valley, but I think that programmers are the key DNA that determine everything else. And I'm a marketing guy. I'm telling you, programmers are more important than marketers. You can quote me. In the Macintosh division, which is where I first encountered this. I mean, that was a engineering driven division. I think, frankly, Apple to this day is an engineering driven company. And so Steve Jobs considered programmers, artists. You know, this concept of, you know, you right. You need to write 10,000 lines of code per day or, you know, whatever. This kind of quantitative management of programming that just was not present in the division. It was all about - we thought we were changing the world, denting the universe, whatever metaphor you want to use. And these programmers were artists and had the art of the Graphical User Interface and how they could do things with assembly language in much tighter code that executed faster. And it was a completely - it was programmer as hero.

[00:07:13]
BH: So how would you inspire developers in other companies today to feel like they are able to change their company, change the world? What do you say to them?

[00:07:22]
GK: You think I know the answer to this question? Like what made you think we had - this was not in the pre call briefing, right? I don't know the answer to that question. You know, I think many companies, especially when you get larger and larger, you how does an individual programmer express his or her art? That's I mean, if you're - you know, imagine if you're writing printer drivers for a Windows clone, I mean, I you know, I don't know how you - oh, my God, Mom, today I wrote

like 10,000 lines of code to help printing in, you know, whatever, when you want to change from Comic Sans to Helvetica. I don't know how they - it's a different world.

[00:08:12]
BH: At least we got Guy Kawasaki to say Comic Sans. That's the main thing. So have you met any particular standout programmers along the way that you still remember that you thought, you know, they are absolutely outstanding?

[00:08:27]
GK: You know what – it's, I wish, it's been a long time. But, you know, there are people like Andy Hertzfeld and Steve Capps, Bruce Horn, who worked on the Macintosh Finder, that's the system software that manages the desktop, and Andy worked on the ROMs. And Bill Atkinson wrote Mac Paint and he wrote QuickDraw, which is the graphic routine's in the Macintosh ROMs. So, you know, I think those people were truly artists, truly geniuses in what they did. It was their, you know, sort of language. It was their platform. And I have to tell you, since then, I, I, I have just not been in touch with programmers that much. And, you know, the good news may be that with these modern languages, programming has become much more of a accessible skill. So it's empowered more people to become programmers. But, you know, in my day, it was ones and zeros with assembly language. I can say - that's how, it was rubbing two sticks together. Yeah.

[00:09:43]
BH: I remember typing in Assembly language from magazines of those days that used to have a whole lot of programs listed and she would literally type it in word. And it was like learning something parrot fashion and you had no idea what it was. And then you got these really strange things happening.

[00:10:02]
GK: Do you remember Byte magazine?

[00:10:04]
BH: Yes I do, yes.

[00:10:05] GK: Oh my God, yeah. Bradley, but you know what the funny thing is, that 20 years from now, people say, "I mean, people used Java and used Python and they used C++, like, what were they thinking?"

[00:10:23]
BH: Yeah, where missing out a semicolon would wreak havoc and take days to find.

[00:10:28] GK: They're rubbing two sticks together. Well, how did they ever create Facebook?

[00:10:36]
BH: Oh, I asked Mark Zuckerberg to come on next time and he can tell us.

[00:10:40] GK: Yeah. Yeah.

[00:10:41]
BH: So what advice do you have for maybe graduates that are thinking about going into programming as a career?

[00:10:50] GK: Well, I think that, I maintain this romantic view of programming that it is an art and, when it becomes just an assembly line… Here's a metaphor I understand much better. OK, so let's make maybe the false equivalency, that programming is like writing, and so writing is something I truly understand, and I truly do believe that writing is an art. So there are people who are very good writers, fiction or nonfiction, and there are people who are not, and there are people who have… write buggy code. And then there are writers who use incomplete sentences and, you know, don't use the Oxford comma and don't put a comma after a conjunction between two independent clauses, and they use the passive voice too much and they take five sentences to what a good writer would say in one. Just like there are programmers who would take, you know, a thousand lines of code where a great programmer would do it in a hundred lines of code or maybe ten lines of code. So I think that to say that all programmers are geniuses and technical wizards and all that stuff is like saying every writer is a great writer. It's simply not true, simply not true. The degrees of expertise are very obvious. Now I can't tell you I can judge that, but I could with writing. But I think you should think of programming as an art. And there are some artists and there are some just… charlatans.

[00:12:33]
BH: I completely agree with your point about programming as an art. It's an absolute creative skill, and it's probably one of the most creative parts of science. I completely agree that.

[00:12:44] GK: Well, I don't, you know, I don't think you would ever say to a writer, you have to write 50 pages today. Now, there is a scenario where you have something like that, so if you are a writer and let's say you're a novelist, you may set yourself a goal of writing. Well, novelists and true writers would set themselves a goal of writing two pages a day. That would be a very good day. So maybe the number was too high. So, you know, I believe a writer can say, OK, every day I'm going to write two pages. A better programmer, a good programmer could have that same kind of attitude. But I don't think it's a thing where management can say, all right, so our most productive programmer, we're going to measure productivity by lines of code. That seems to me crazy, that - not all lines of code are created equal.

[00:13:36]
BH: Yeah, and if anyone is actually wondering whether what guy is talking about with lines of code have a look on, I think it's Wikipedia. And look at KLOCs – K, L, O, C - and you'll see that. Developers really were assessed on how many KLOCs they could they could –

[00:13:54] GK: Really?

[00:13:55]
BH: Yeah, it's amazing. I'm not going to mention the company. Everyone can read that for themselves on -

[00:14:00] GK: Does it have three letters?

[00:14:04]
BH: Yes. And is related to HAL from 2001 but I'm not going there -

[00:14:10] GK: Continuing in the analogy of writing and programming, obviously. Or maybe not, obviously. But I am a writer and I will tell you that there was a book that was a catalyst for me becoming a writer, and it's called If You Want To Write. And the author is Brenda Uelend, U, E, L, N, D. And so this is a book about becoming a writer, believing you're a writer and just freaking writing, not listening to people tell you that you need a Ph.D. in literature or English to become a

writer and you need to take a course and all that - basically Brenda is saying, if you want to write write, don't listen to anybody else's estimation of your ability. It's a creative process. So now you maybe wonder, what the hell does this have to do programming? So I think, you know, if Brenda Uelend was a programmer, she would have written a book called If You Want To Program, and she would say, you know, if you want to write software, write software, you know, you don't need to get a Ph.D. in computer science. You don't need to, you know, go take the dot net class or the JavaScript class, just learn it and write. And I would make the case that anybody listening to this who's interested in programming, you would be emancipated by reading If You Want To Write by Brenda Ueland.

[00:15:24] GK: Thank you. Okay. Well, look like always Guy, it's been a pleasure having you here. Thank you so much for joining us on this special occasion. To all of our listeners, I hope you had a great time with us today. Please make sure you subscribe and join us again next week for another episode of Tech Reimagined. Thank you.