cps-e24-alec

Sat, Dec 10, 2022 8:29AM 19:41

SUMMARY KEYWORDS

python, ci, circuit, project, pi, pcb, files, code, pretty, github, continuous integration, moving, adafruit, library, people, tools, boards, run, alec, create

SPEAKERS

Paul Cutler, Alec Delaney

P **Paul Cutler** 00:01

Welcome to the circuit Python Show. I'm your host Paul Cutler. This episode I'm joined by Alec Delaney. Alec is sponsored by Adafruit to work as a part time developer on the circuit Python project. He works on the Python libraries as well as the continuous integration across the organization. He's also a maker working with circuit Python to create robot friends and currently works full time as a mechanical engineer for a medical device company. This episode is brought to you by PCBWay. With over a decade of experience PCBWay is one of the most experienced manufacturers in PCB prototyping and design. Whether you're an engineer students or hobbyist PCB way offers a simple and fast prototyping service. And it's cost effective at only $5 for 10 PCBs and check it out PCB way.com/project, where PCB way helps makers and hobbyists collaborate on their designs and projects. Make your design a reality and check out PCB way.com For all your PCB needs. And they also now offer CNC machining and 3d printing services, visit pcbway.com For more information, thanks to PCBWay for their sponsorship, Alec, welcome to the show.

A **Alec Delaney** 01:05

Thanks for having me.

P **Paul Cutler** 01:07

I'm glad you're here. Tell me how you got started with computers and electronics.

A **Alec Delaney** 01:10

The first memory I have really of programming on a computer actually was I think back in grade school it was with JavaScript. So I started off by know a lot of people start off with like C or you know, whatnot. But I started off with it with a pretty dynamically typed language. So that's how I got my start. And then I went to school for engineering, but I actually went for mechanical, but

one of the courses we had to take was a kind of like a programming course, where we had MATLAB, which is a language used pretty much a lot by mechanical engineers and the electrical. And then the other one was actually c plus plus. So that was kind of my second language. But then being in a field sort of next to hardware and data collection, and therefore data science, Python came up pretty quickly thereafter. And so I found Python and then being in in hardware, when working knowing Python, it was only a matter of time before I stumbled upon circuit Python, and I fell in love immediately, just to be able to have an idea and and know Python and you know, get started on a microcontroller to remove that barrier of, of knowing how to compile or set up a tool chain, it just it was it was it was magic to me.

P Paul Cutler 02:28

You're sponsored by Adafruit to work on continuous integration using GitHub actions for a number of the circuit Python repositories. For those that might not know what is continuous integration or CI,

A Alec Delaney 02:40

continuous integration or continuous development is the process of basically removing the manual steps from parts of the code cycle. So of course, there's development and there's there's writing code, but there's all the other parts like testing and, and shipping, that needs to happen to kind of create good code and in best practices. And so what CI does continuous integration is the process of constantly doing those, right? So instead of, you know, asking people like, Hey, do you mind testing that code? Or hey, do you mind zipping it up, and you know, uploading it, it does it automatically, basically. And so it's that process, just really, you know, those are just some parts of the CI for the circuit Python project and does Adafruit uses in general is, is pretty varied in what it does. But it all does that basically, it all automates these, you know, sometimes critical processes behind the scenes so that everything just kind of works seamlessly.

P Paul Cutler 03:40

So tell me about some of those things that are behind the scenes that, you know, the average user doesn't see that makes all of this happen to bring circuit Python to that user.

A Alec Delaney 03:50

So the CI that Adafruit uses for circuit Python, and really the whole circuit Python ecosystem. Some of them are kind of where you might expect in that traditional side of things, right. So for example, one important job the CI does is for actually taking all of the library code, and kind of bundling it live quite literally, you know, for the circuit Python bundle, and you know, creating that bundle and uploading it to various places, for example, circuit python.org, it handles uploading it to there also handles creating GitHub releases. So likewise, it helps put it on pi pi. So it does all of these more traditional things that you would expect a CI to do. Again, you know, there's a whole bunch of infrastructure for testing code, all the code comes whenever you create a library or or, or submit a change to a library, all all the formatters, a linter runs all

of these very traditional CI things. But what's really cool is that we actually use the CI If for some, for some other things that are, you know, I would say are pretty important, even if they aren't the most traditional use, for example, the weekly meeting that's held every Monday, the meeting notes for that are actually generated on by the GitHub actions CI, which is really cool. Because a lot of the stats were reading, you know, how many pull requests how you know, how many milestones you know, who are new contributors, right, that all that information is GitHub information. So we just were able to take that, and run it through the CI and get that, you know, almost natively if you think about it, and then create this document, and you know, and upload it to AWS. And so that whole process is completely automated. It's actually automated daily. So at any point, you could see that information, we only read it once a week. But yeah, you know, I think that part of the cool thing is that we use CI pretty regularly, pretty widespread across the whole project.

P Paul Cutler 06:00

How does the continuous integration make it easy for developers and or the users?

A Alec Delaney 06:06

Yeah, that's a great question. I would definitely say that, you know, for sure, it's probably felt most palpably by developers, or even just contributors, in general, you know, especially with those traditional things. One of the most important things as a developer when you're writing code is that, you know, when you submit it for a pull request, other people are going to be looking at it. So you don't you not only want to make sure that it's it's good code, but that other people can follow along. And so of course, that means things like commenting and whatnot. But part of that, for example, is readability. So part of what the CIS job does across all the libraries is formatting right? So we have we use black it is, if I remember, correctly, the tagline is the unrelenting for matter. It is it pretty much standardizes, what code will look like, if I ran black on the same code as you we should get the same thing. So what's great about that is that when I write code, and other people are going to look at it, I can focus on really what it should be doing. Right, I focus on, you know, is it doing what it needs to do, and then kind of throughout the process, or even at the end, as well, I can have all these automated tools run and say, Okay, I'm gonna move this over here, it seems more readable. Or I can have the linter say, Oh, hey, I noticed you aren't using this variable, we should remove it. And so I can focus on writing the code. And anyone who's contributing can do the same. I would say for the users, it's definitely the more not fringe, but the the other uses of that, when we talk about how people load code onto the Feather M for Express, which you know, can hold, I think it still can all of the libraries. And so that's where the bundle comes in. That bundle is created by the CI, from the user's perspective, it's as easy as, you know, going to circuit python.org, and just picking up the zip file, but behind the scenes, there's a whole, there's the infrastructure to commit the zip file. And before that there's infrastructure for making sure if someone updates one of the repositories that that gets put in the bundle, and, you know, you can go you can go up the chain on for a long ways from there. But, you know, in the same way that circuit Python makes, you know, editing code really easy. The whole project and Adafruit in general uses CI to just make the whole development and user perspective a lot easier.

P Paul Cutler 08:29

And all of this is done using GitHub actions. Yeah. For the

A Alec Delaney 08:33

circuit Python project. And in Blinken related things, I know that GitHub actions is used. I think before that they used Travis, that was a popular CI back in the day. Yeah, I think I came I think even the Arduino library, still use it. But yeah, everything is pretty much all GitHub actions. And, and that's its own ecosystem of cool tools and whatnot, one of the projects that's on my plate now is a lot, a lot of that has to be updated. So I am actively taking a look at it that all over again,

P Paul Cutler 09:09

one of the other projects you're working on is moving packaging to use the pipe project that tomo files, what are some of the challenges in moving to that system.

A Alec Delaney 09:16

So one of the biggest challenges of that is honestly just how to do it. You know, the task is just one of those things where the task is really simple. You know, a lot of the information is already there, and what was setup.pi so the previous file that that had all of the information for, you know, pi pi and if you installed locally how to do that, but moving that to a new file format is tricky, because you can't just blank it, copy things. So you can't just copy paste things, of course, but you also can't just use it's hard to use a template because some of that information is specific to those files. So some repositories for sample have keywords so that if you're on pi pi, you can search for them. So those are specific to each file, the real challenge, there was a blend of how do you use templates. And there are some tools like for remember correctly, format patch, so you can make a commit in one git repository. And git can kind of try to apply it to other ones. And that's actually something ADA bot does, which is really the mastermind developer tool for helping with some of these things. But it's hard to do that here. Because some of the things, you know, you can't do that with keywords, for example. So it took a lot of manual tooling behind the scenes, along with the CI to kind of create a hybrid solution that could you know, template these files and upload them. And then of course, there's the whole tooling for getting it, you know, checking it and making sure that everything went went correctly, one of the classic failures is, and I still have to remind myself that they aren't the same is their Python in pack in the packaging world. There's modules, which are these, like single files, and then packages, which are a folder of files. And from a user's perspective, they're, they're entirely the same. But when you're writing these files, it's really important to tell the file, which it is, and if you get it wrong, it may not be clear, and and it's sometimes even slipped through the CI. So you know, how do you how do you check things that that sometimes can silently fail? So it was all around just a conglomerate, or a conglomeration of different problems with with unique solutions. So it was definitely one of my favorite little tidbit challenges. So far,

P Paul Cutler 11:42

what are the benefits to using PI project that.toml instead of setup.pi

what are the benefits to using PI project that total instead of setup.pi.

A Alec Delaney 11:46

So pi Project dot Tonle, is a configuration file for PIP. So setup.pi is an executable file like any other Python file, and the way it works is you import setup tools, and you give this setup function, all the information you need. And then it builds. The issue with that is you already need setup tools to use it. And so it kind of creates this chicken and egg problem where you need setup tools to call that setup function, which in turn has a setting in it that you can tell it what dependencies you need. But in order to do that, you already need it. And so that was the major problem that moving away from setup.pi to Pi project at tumble solves. That wasn't necessarily an issue for us, a lot of builds come with setup tech tools, it might even be default with Python, which is how it gains such a foothold. But there are other build packages or build tools like flit is another one. And they use pipe project ensemble for that same reason where they specify what they need to build. And then PIP will go install that and it will do that in a virtual environment, and then build the package and it'll build wheels, it'll build a source distribution, it'll do all of that. That isn't to say that setup.pi doesn't have a place anymore. I know, one of the major places that I see a lot still is with the use of Cython. So Cython is a language that you can write. And it basically is kind of Python like and it gets basically compiled down to C code like any other Python C extension, and then that gets compiled like any other Python C extension. But in order to do that, you need to do a couple things in the setup.pi file. So moving things to project automall at least gives them the chance to say hey, like these are the build dependencies we need. And in that you can even specify Cython. That's definitely the predominant reason. And then it also gives us the ability to if in the future, we want to move away from setup tools to something like flit or poetry that both us pipe project ensemble and gives us a better way to do that, now that we already have that. And to be fair, I know that Blinka, for example, still needs to set up that pi. So again, as an executable file, it's able to do runtime dependency. So one thing that Blinka does, when you install it, it checks for specific libraries that are only available on the Raspberry Pi. So what it's able to do is as you're installing it can say hey, are you a Raspberry Pi great, here's the libraries I'm gonna throw in as a dependency. And if not, then it doesn't doesn't do that. Even on our own projects. There's still a place for for setup.pi.

P Paul Cutler 14:56

Switching gears I wanted to ask you about one of your personal projects the circuit Python, new Kia, tell me about that.

A Alec Delaney 15:02

I had for a while been wanting to make a project that I could share with people. So a lot of my robot friends that I had made, you know, when I'm not working on the CI, I love working with the circuit Python boards themselves, I wanted to make a project that wasn't hyper specific, like some of my other ones had been because I wanted to share it. And one of the things I've really enjoyed about being, you know, working with Adafruit and just working with open sources in general, and some of your other guests have mentioned is how great it is and what you can learn from from people in the community. And there's a lot of great makers that do open source hardware in it. And it was really inspiring to me, and I saw I wanted to do something that

wasn't hyper specific, and I wanted to do something that I could share. And so when I was thinking of things, I think one day I had come home, you know, you know much around this time when you know, up in the northeast, it gets dark so early around the holidays, and I and I was thinking, you know, oh man, like I'm gonna be late to light candles for my for my new Kia or you know, you may say a menorah. And it kind of hit me that I was like, I could probably automate this, like I could probably come home, and it could be lit. And so it kind of like what started off as this neat idea of of having that happen. Turned into this project. So the circuit Python Rukia is exactly that. So it's a custom PCB, which was fun, it was my first PCB I've made. And what it does is it's got LEDs on it, and it runs on a cutie pie s two, and it's got the LEDs, it's got a piezo electric buzzer on it. And what you do is you plug it in the cutie pie s two connects to the Wi Fi, and it looks up the date it looks up when Hanukkah is and then it says you know, either Hey, it's Hanukkah and it lights up the correct candles and plays a little tune or it will wait to do that. And what's really exciting is like just it was the first it's been the first project that I've done in open source that I went and got open source hardware certified. So I went and got it and now as it's I just reordered the boards with its certification mark on it. And I'm excited to hopefully if the holidays slow down a little bit in how fast they're coming. I'll try to get those out to people before Hanukkah starts.

P Paul Cutler 17:26

What a great project. My last question for you. You're about to start a new project which board do you reach for?

A Alec Delaney 17:32

Yeah, I'm definitely biased after the circuit Python Yukiya project, I would definitely definitely have to give two answers. I think that any of the RP 2040 boards especially with you know chips, Chip shortages and whatnot, how available those have been and how cheap they are, is it makes it just just just a fantastic chip to work with you know knowing that I can grab any one of the circuit Python boards that has the RP 2014 and start you know hacking away and not have to worry about too much is great. And how on top of things they are a you know shout out to gentler who with the PI cow pow to how great that is the any chip now it has Wi Fi built in right so now you can use the RP 2040 and great you have Wi Fi included but I also do there's there's a special place in my heart not just from this project but more from my mechanical side where when I'm thinking about you know how I'm going to make things and make things small, especially the cutie pie format and specifically I've really enjoyed s two just knowing that if I'm doing something small or I want to you know there's a lot of constraints and I only need a few inputs and outputs anyways just knowing that I can I can make something and add that to the board really quickly and and you know have a lot of flexibility on just one tiny little format is just a joy.

P Paul Cutler 18:54

I'm with you my current projects are using the ESP 32 s two as well and it's just a great chip to work with and having Wi Fi is just changes the entire game.

A Alec Delaney 19:04

Absolutely I figure if the if the circuit Python Nicaea you know perchance doesn't happen this year. I've already made the order and I already have 15 Cutie Pies here and you know if they if I have to use them for other things, and I'll well Soviet,

P Paul Cutler 19:19

I'm sure you'll find a use for them. Alec, thanks so much for being on the show.

A Alec Delaney 19:23

Thank you so much.

P Paul Cutler 19:26

Thanks to Alex for being on the show. That's a wrap for season two. Thank you to everyone who has listened in the show. We'll be back in early 2023. Until then, stay positive