cps-17

Sat, 8/13 1:15PM 16:09

SUMMARY KEYWORDS

micro python, python, game, robot, display, people, easy, sensor, programming, bit, computer, microcontroller, buy, bouncing, problem, simple, super, code, memory, project

SPEAKERS

Paul Cutler, Radomir Dopieralski

P **Paul Cutler** 00:02

Welcome to the CircuitPython Show. I'm your host Paul Cutler. This episode, I'm joined by Radomir Dopieralski. Radomir is a Python programmer by day, a walking robot builder, game developer and wiki enthusiast by night. Radomir, Welcome to the show.

R **Radomir Dopieralski** 00:17

Thank you for having me.

P **Paul Cutler** 00:18

Glad you're here. How did you first get into computing and electronics?

R **Radomir Dopieralski** 00:23

Well, I was right behind the Iron Curtain. So we didn't really have home electronics, much in the space, there were other priorities. But what we had was arcade games, they came pretty quickly, of course, smuggled. So the first computer I actually saw was an arcade cabinet on some shady, shady, gone somewhere next to the train station. At that moment, I was hooked. I never actually played those games, but I could spend like hours standing there watching other people play. As soon as I got access to any computer, I tried to write those games. Well, no, those games that somehow were similar to what I saw arcades, fortunately, when my parents bought a computer, because it was a very old one. And most of the games you could buy at the time didn't work on it anymore. So I had to spend a lot of time getting them to work, I didn't have my hard disk drive. So I had to pack them all the files on as many diskettes as necessary. And you know, there was a lot of trial and error in there, I learned much more than if I spent that time just playing games. And one thing came to another I went to it at the university and so on. So I'm a programmer. Now,

P Paul Cutler 01:43

how did you discover micro Python and circuit Python?

R Radomir Dopieralski 01:46

Well, I discovered micro Python from the micro Python Kickstarter back when it just started, I thought it was a great idea at the time, I was mostly using Python, to work and for my hobby. So it was a great idea to have it on device like that. And of course, the most immediate idea, or you could use it for was to have a handheld gaming device with problem was that it was a bit expensive at the time, you couldn't you know, design a handheld device that would cost so much that nobody would build it. Except for you. So that didn't work. For me. What really got me started into micro Python and got me contributing to it was the second Kickstarter with the ESP 260 X version, because at Ritalin port, that was still relatively public, powerful 80 megahertz CPU, so you can do a lot with that, that got me on my way to, to making a handheld device that you could program with micro Python.

P Paul Cutler 02:50

So you've been working with a number of handhelds over the years that you call the pew pew family. Tell me about that.

R Radomir Dopieralski 02:56

Right. So apart from my quest to actually recreate the arcade games, or like handheld games from from Gameboy or from swig, I also got involved with in the Python community into doing workshops, and was very annoyed that the first half an hour an hour, often, at the workshop you spent helping people to install everything helping people to install Python, have been paid people to install PI game if you're arguing against they making. So I wanted a solution that would let me just tell people, okay, take this device connected to your computer. And we are starting with the game making. So I needed like the minimal viable gaming console, I could use to teach programming. And that's why I came with PPO. And since it's for workload, so you're likely to have to buy like 20 of people, I had to make it as cheap as possible, as affordable as possible. It's just an eight by eight led LED matrix and the smallest chip that can run circuit Python possible. But since it's still good Python, you really, it's super simple to start, you just connect it, it comes as a USB drive and write your code. So that's that's it? Well, if you want to see the error message, that's a bit more advanced, but most of the time people manage that.

P Paul Cutler 04:27

So they get a pupi device in a workshop. What do you teach them? What did they learn?

R Radomir Dopieralski 04:32

So the library I use for, like, displaying things on this display and for handling buttons. It's very minimal. Because I really wanted to get down to the main game loop. People have suggested to me, okay, make those callback functions and update on button press, like in the microbeads right where you can. It's much easier for kids to write code like that because they just have to fill the gap. apps, and everything else is there already for now, I really didn't want that. I wanted to let people own the device and really struggling with every single problem, like starting off of reading an interactive application. So I don't even do the bouncing on the button, you have to do the bouncing yourself, if you want to have like a turn based game where one key press is really one key press, not multiple key presses, you have to do the bouncing yourself, you basically only have a simple function for writing a pixel to the display and some functions for copying images in the memory. And that's it. Since the games are really simple. I get away with that. But I think that you can really teach things that are very much transferable into other like programming problems, like suddenly you have to program a robot. And that's also an interactive programming, you have to react to sensors, you have to move the robot in real time. So you get the same kind of problems as you learned by writing writing games. So it's not just about the games actually, doing not if you you know you learn to write again, for pure pure, you want to be able to next day to write quake. At least you get this idea about how interactive applications work inside.

P Paul Cutler 06:21

How does circuit Python make it easy to program games?

R Radomir Dopieralski 06:24

Well, as I said, you're connected to the computer comes, it comes up as a disk drag. And every time you save it restarts and runs your game from the scratch so you can see from scratch so you can see what changed. That's basically it. Otherwise, well, I tried to write the library in a way that makes it easy, there isn't really much you can count it's cope with the more complicated game consoles where you have actual graphics, it's also super easy with the disk because you can copy the files onto the console all the assets, or the sound, you already have the LTO IO module. So you don't have to figure out how to make sound mudras make wav files and copy them. A lot of reading modules, there is a keypad module that does the bouncing for you and handling of the mountains, there is the there is a whole display I O that lets you do more advanced graphics than you can do with my libraries. So there is a lot of stuff already done for you. Of course, if you want to do something really strange or new, you will have to do it yourself.

P Paul Cutler 07:40

What are some of the challenges programming games and circuit Python,

R Radomir Dopieralski 07:43

those are microcontrollers. So they clearly have very little memory, it's like programming on those eight bit microcontroller computers at home. If you have like 240 pixels by 240 on your display, and that's 16 bit car because the display only super 16 bit cars that two bytes per

pixel. That's really a lot that's like kilobytes of memory Sanan. So you can even store all of that in microcontrollers memory, you have to do tricks. So you have sprites and ties by repeating the same image multiple times, let you know, treat them you only store this one small image it gets it gets repeated. So it uses much less memory. With sprites, you can move things around your screen without having to, you know yourself clean the previous position and read throughout the whole thing in the new position. So you don't have to keep the background in your frame buffer. There are also things like on disk bitmap in circuitpython. That let you draw bitmaps directly from the flash from the USB drive without having to load it into the memory. So that that helps as well. But generally, depending on the device, of course, if you have an ESP 32 s two Rs three, with four megabytes of PS RAM, then you can do a lot more than you can do on Sunday 21 with 16 kilobytes of free RAM.

P Paul Cutler 09:24

If I quote a game using a certain display, are all users going to be locked into using that specific display?

R Radomir Dopieralski 09:31

Well, it depends how you do it. The display aisle lets you track the size of the display for instance, inside your game so you could scale your game depending on what size the displays and if you do it like in a naive way, just for this one display. Probably it's going to be difficult to scale. Since there is no game framework right now. procedurally Python, I think there should be at some point, right now you are left to figure out those things yourself. So you have to think ahead and write your code in an elastic way to be able to target multiple devices. Yeah, it's something that you learn after your second or third game, that you have to think a bit about how how the elements of the of the interface are going to move if the display changes.

P Paul Cutler 10:29

One of your many projects is building out robots. Can you tell me about the fluff bug?

R Radomir Dopieralski 10:34

Actually, that's what got me into electronics. Initially, I decided, oh, one day I, I looked at all those Arduino projects. And I said, Okay, I want to make a working group, a four legged spider working, how hard can it be, I got like a servo controller, not even one that you could program, but one where you could record server positions, and battery and a bunch of service. And I connect it all together. And turns out that working is a little bit more complicated than just recorded positions. Then I switched to Raspberry Pi on leads and a bigger battery, then I switched to standard service, then I switched. Once micro Python was out, I switched to Oh, before micro Python was out, actually, I switched to ASP 266 That was communicating with a computer where micropile were normal Python was running and sending the commands to the robot over Wi Fi. And then my micro Python came out. And that made things much easier for me. But still, at some point I I was because I wanted it to also be super cheap so that other people can build it, and have it use it as a starting point for their projects, like, you know, make

it look like some robot from a game or moving a day like so there is this called t o t E project where I'm still using Arduino because it was the cheapest boards you could use, where I have like step by step instructions for for building the robot. And it's controlled with a TV remote. So super easy. And now I'm working on an even cheaper, even more versatile and even easier to program robot with circuit Python, since the main cost of the robots made of the mechanics, or the servers themselves, the original robot had 12 of them, that comes up to like $25, just for the servers, if you want to buy them. So the current one has eight, and I have compromised I will be turning I can no longer like 13th place I have to turn like a tank that because of that, because I can't move the next sideways anymore. But you know, it shaves off like one 1/3 of the price from from the ramp up. So So I think it's worth it. It doesn't really have a function. It's it's a desktop robot for walking for programming, it's yourself, you can make it dance, or you can make it you can put sensors on it, it takes shields that actually attached like face masks, because most of the sensors you want to be facing forward, like a distance sensor or issue sensor. So it's like a face mask. And I also made one that display. So you can show a face on it or an eye to make it look super creepy.

P Paul Cutler 13:37

After I get a picture that from you and put that in the show notes.

R Radomir Dopieralski 13:40

Yeah, so that's that's there. The challenge for them is the code. Like the mechanics, I have pretty much figured out. It's the same for several iteration. Now. The problem is with the codes, because as I said, it's an interactive application. Right now I'm struggling with the I think I think I owe and I think in general, I want to rewrite all the competition that currently does only one thing of the robot. So either you have a working robot, or you have a robot with the AI, or you have our own sensor, I want to combine it all into one code.

P Paul Cutler 14:17

So if you're using async IO, you can asynchronously program so the eye is moving while it's walking, for example. Exactly. That's a really neat project. I'll make sure I link to those in the show notes. We're almost out of time. But the final question I have for you is you're about to start a new project, which microcontroller Are you going to reach for?

R Radomir Dopieralski 14:36

The practical answer right now is Sunday 21. And that's because when the chips shortage started, I bought the last 50 ones from my mother. So I have them in my drawer right now and they are super easy to use because you literally only need two capacitors for them. And not and not even like if you're running from a battery you don't Even the voltage regulator can connect the battery directly to them to capacitors that that's it. So I have a lot of simple products that that use that chip just because I already have it and I know it and it's convenient. What I should be doing is using the ARP 2040 Because that's the only tip you can bind right

now. Basically, are the expressive ones. I'm still a bit intimidated by the fact that you need a crystal you need external flash memory. You need all those stuff around the trip, you have to design the recipe for it properly. So I hope to get over that hump and start using it.

P Paul Cutler 15:46

Those are great choices. Radomir thanks so much for being on the show. Thank you. Thank you to Radomir for being on the show and I apologize for any mispronunciation. For shownotes transcripts and to support the show visit circuitpython show.com. Until next episode, stay positive