Kia Takatū ā-Matihiko
Digital Readiness

Pīkau Name: What is programming?

Video Name: What’s the best programming language (EMP07-6)

Presenter: Professor Tim Bell, Joanne Roberts

Joanne: Tim, I’ve been working with you for a while now and it seems that every second week, I hear of a different computer programming language, all used for different kinds of things. How do I know which one I should be teaching in my classroom?

Tim: The answer to that is quite a detailed one, but one thing I will tell you, someone will always tell you that you are using the wrong programming language! What's a good programming language for education is different to necessarily what's a good one in industry. It's probably good for us to explore some of the criteria you would use for choosing a programming language. It's a bit like when you teach a student to drive, you know, you’ve got kids that learn to drive?

Joanne: Yes, yes.

Tim: So the question is “What's the best car to teach someone to drive in?”

Joanne: The one you can afford (laughter).

Tim: Ok, thats a good criteria.

Joanne: Nice and small (laughter).

Tim: Nice and small, good insurance policy on it maybe? You might think that professional drivers drive trucks and buses and bulldozers and things like that, but you wouldn’t really want to start there. It might be motivating for a student, but actually it could be very demotivating. I mean you are trying to get a truck into gear and it won’t work and all these things, and so you want something nice and simple.

Joanne: Yes, because what you learn in that cheap car, my daughter is learning to drive at the moment and what she is learning in the little car is exactly the same skills; well not quite exactly, but they are similar skills to what she needs know to go in a truck as well, or anything else.

Tim: Exactly, and you know as you teach her, that you are teaching her the general skills that are going to be useful even though that particular car, the moment she hops in a different car she will probably go "Oh you didn't teach me how to use this other car". And of course one of her friends will tell her "That's not a cool brand of car, you should be driving this brand of car" and things like that. That can be a consideration, just the fact that a student sees something as cool, but there's a few things that teachers need to look at when they decide what programming language to use. I think one of the first ones, is "Is it free?" and by free, not just doesn't cost money - that's important - but also just freely available. If you don't have to pay for it, it means that if a student goes home they can get a copy for themselves or just use it.

Joanne: They can share it with their people at home, they can "Hey, check out what I did today at school" and there are no issues with that. That makes sense, so free is a good thing to be.

Tim: If it's easy to get, maybe they don't have a computer at home, but they might go to the library and the library's not going to suddenly install a new language for them necessarily, but if it's available through a web page and things like that. You want to think if a student gets really excited, can they use it elsewhere?

There are other criteria: as a teacher you need to think about the age of the student and what motivates them. For younger students what sort of things, I guess, would you be worried about with them?

Joanne: They like to have fun sounds, its a good thing for them. How much language they need to be able to use for it. If they can't get their full stops in the right places in their written, normal writing work, they're probably not going to get their punctuation correct in a programming language.

Tim: Actually I remember one class I had of young students, I think we spent 10 or 15 minutes just typing in the password to log onto the site and things like that. So that's where the block based languages are really good, because they just need to recognise it rather than have to type it. For an older student - and if you think of an adult student who goes "Hey, I'm going to take six months out of my career and learn how to program" - they'll want to use something that's going to be ready to go, and they don't mind working hard. Their motivation is extrinsic - its like "I don't care how hard this is, it will get me a job". So that's another consideration.

Joanne: I also like to look and make sure that if its a language that I am going to be teaching my class, what other supports are there? Is there something that if a kid asks a curly question, I can go out and find the answer, or if they are at home, they have been enthusiastic taken it home and they have a question, can they look up the answer themselves? So what resources are there available online or through their friends or things like that?

Tim: Hmm, and supports for you as a teacher and supports for the kids, and also I think one of the really important supports is the person in the next classroom, possibly. There might be someone in your school who is kind of quite comfortable with this stuff, and if there is a language that they are comfortable with - or a system - it may be quite a good one to go with because you know that whatever goes wrong, you've got someone on the spot to help you with it. I think that also applies as a teacher; some teachers have been programming for a while and there's languages they are comfortable with; that's a good one to consider just because if *you* are feeling comfortable, students aren't going to get the message from you that "this is confusing and I'm trying to figure this thing out". Now the reality is for many people, they will be figuring it out as they go along and that's fine and the key thing is that you can learn with your students as well. Just thinking about where's the support around for this? There are hundreds of programming languages out there, there's are probably 20 or 30 that are quite widely used commercially, but in education, well there are a lot that get used in education but there's probably 3 or 4 that are really common, so inevitably if you keep bumping into the same ones, they are going to be safe ones to go with.

Joanne: One thing I have to remember is that if anyone is giving me advice as to what program I should be using, I have to remember that I know my students best, I know what they need to learn, I know what I'm comfortable with, I can make that decision myself.

Tim: That's right, and it depends on the cognitive level of your students and what's going to motivate them and so on. I did use the analogy with training wheels on a bike, but it's also like streamers on a bike. You know, for a young kid, having streamers is so cool, and for an older one it's so embarrassing. There are all those things to take into consideration.

I think the other thing is that when we are teaching programming, we need to talk about, "I am teaching you how to *program* using, say, Scratch, or using Javascript" or something like that. Again, going back to the car analogy, rather than saying "I am teaching you Toyota" or "I am teaching you Holden" or something like that, we would say "I'm teaching you to drive". Now, when the student switches to a different car they might feel that you haven't taught them the other one, and it's true that there may be a lot of conversion to do, but I think it's really important to have that language - that a computer programming language, it's not just a piece of software, and another one is a different piece of software, there is a whole principal you are teaching there; the elements of programming. You happen to be doing it using a particular language.

Joanne: Hmm, even though my girl is learning in a smaller cheaper car, those skills she is learning now, they will relate to driving a formula 1 sports car if she wants to in the future.

Tim: That would be very cool!

Joanne: Um huh (laughter).