THE MAGAZINE FOR COMPUTING & DIGITAL MAKING EDUCATORS

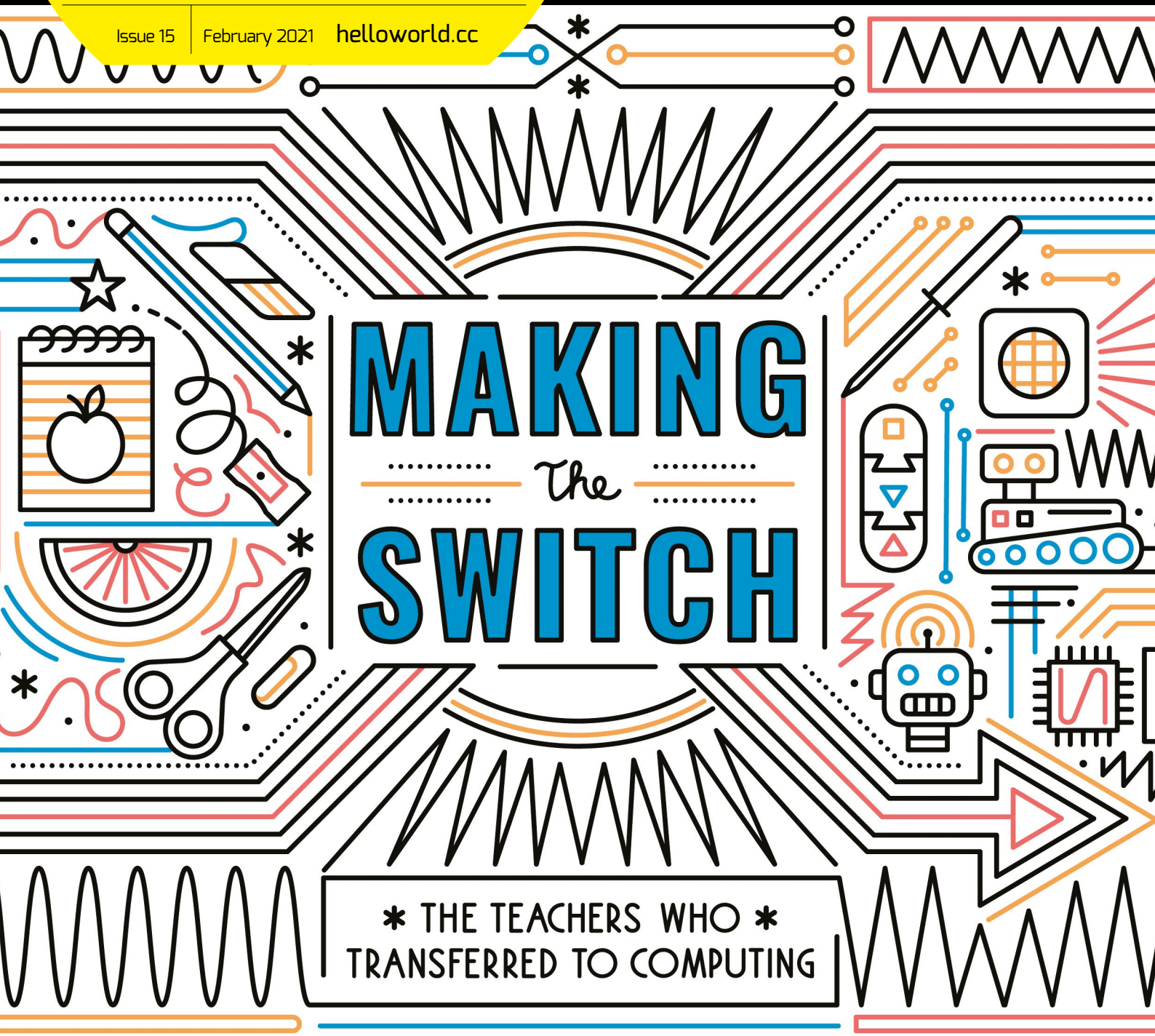
# (Hello World)

Issue 15 | February 2021 | helloworld.cc

**ANNE-MARIE IMAFIDON**
The Stemettes CEO on equity in computing

**PRIMM AT PRIMARY**
Get younger students talking about code

**FLOW STATE**
A frame of mind for learning



# MAKING The SWITCH

* THE TEACHERS WHO * TRANSFERRED TO COMPUTING

**PLUS** GCSE OPTIONS ADVICE • METACOGNITION • UNIVERSAL DESIGN FOR LEARNING • ASTRO PI COOLEST PROJECTS • LOGGING ON IN YEAR 1 • PHYSICAL COMPUTING IN THE CLASSROOM ARDUINO APP • RETRO VIDEO GAMES • DIGITAL FOOTPRINTS • BINARY MISCONCEPTIONS

Raspberry Pi

# Learn with the Raspberry Pi Foundation

Free for everyone anywhere in the world

## Teaching resources

Discover training, resources, and guidance to help you teach computing with confidence.

**teachcomputing.org**

## Project library

Browse our 200+ online project guides that include step-by-step instructions for learners.

**projects.raspberrypi.org**

## Digital making at home

Join our weekly live streams and take part in Coolest Projects and Astro Pi Mission Zero from home.

**raspberrypi.org/learn**

## Support for parents

Watch our support tutorials and access engaging resources for your child.

**raspberrypi.org/learn**

Raspberry Pi Foundation, UK registered charity 1129409

# HELLO, WORLD!


**Computing is different to most other school subjects: it didn't exist in anything close to its current form when this generation of teachers were at school themselves. This, combined perhaps with the disparity in salaries between software engineers and teachers, means that there simply aren't that many computer science teachers with computer science degrees. And it's not a problem unique to the UK: the nonprofit Code.org found that in 2017, there were only 36 teachers in the US who graduated from universities with computer science degrees, compared with 11,157 maths teachers and 11,905 science teachers.**

In order to make up for this shortfall in computing teachers with a background in the subject, schools have been encouraging their staff to switch specialisms. My colleague Amy O'Meara and I wanted to find out what it was like for the teachers who made the switch. Why did teachers choose to change subjects? What were the challenges they faced? And were they happy with their decision? We share some of their stories on page 20.

This issue, we're continuing to share some of the feedback from our monthly reader surveys, as well as some longer reader interviews we carried out at the end of 2020. It's invaluable to hear what readers find useful and how we can improve the magazine. This is my last issue as editor, but if you have ideas to share in the magazine please do let the team know. They look forward to hearing from you at **contact@helloworld.cc**!

Sian Williams Page
**Editor** (@swilliamspage)

## FEATURED THIS ISSUE

### CHARLOTTE SPENCELEY

Charlotte is a Key Stage 2 teacher at Giffard Park Primary School. She shares her experiences of incorporating tablets into her teaching on **page 36**.

### NIKHIL HANDA

Nikhil is a software development engineer at Adobe Inc in Faridabad, India. On **page 39** he describes what it's been like running a CoderDojo during the pandemic.

### FELIENNE HERMANS

On **page 31**, Felienne – associate professor of Computer Science at Leiden University – introduces Hedy, a new programming language that introduces syntax rules gradually.


(HW)

## EDITORIAL

**Editor**
Sian Williams Page
sian@helloworld.cc

**Assistant Editor**
Amy O'Meara

**Subeditors**
Louise Richmond and Amy Rutter

**Subscriptions**
Joshua Crossman

**Social Media**
Lizzie Jackson

## DESIGN

criticalmedia.co.uk
**Head of Design**
Lee Allen

**Designers**
Sam Ribbits, Ty Logan, James Legg

**Photography**
Raspberry Pi Foundation, Adobe Stock, Alamy

**Graphics**
Rob Jervis

**Cover**
©Muti, Folio Art

## CONTRIBUTORS

Richard Amponsah, Phil Bagge, Vanessa Bakker, Simon Baldwin, Matthew Barr, Richard Bateman, Helen Brant, Katharine Childs, Valentina Chinnici, Josh Crossman, Jonathan Dickins, Diane Dowling, Anna Doyle, Catherine Elliott, Rebecca Franks, Ben Garside, Dave Gibbs, Katie Gouskos, Sway Grantham, Nikhil Handa, Felienne Hermans, Tanya Howden, Anne-Marie Imafidon, Kevin Johnson, Hayley Leonard, Sandra Maguire, Francisco Mireles, Amy O'Meara, Ross O'Neill, Alan O'Donohoe, Eleanor Overland, John Parkin, Emma Posey, Dahlia Poulton-Trask, Neil Rickus, Chris Roffey, Marc Scott, Sue Sentance, Charlotte Spenceley, Victoria Temple, Sian Williams Page

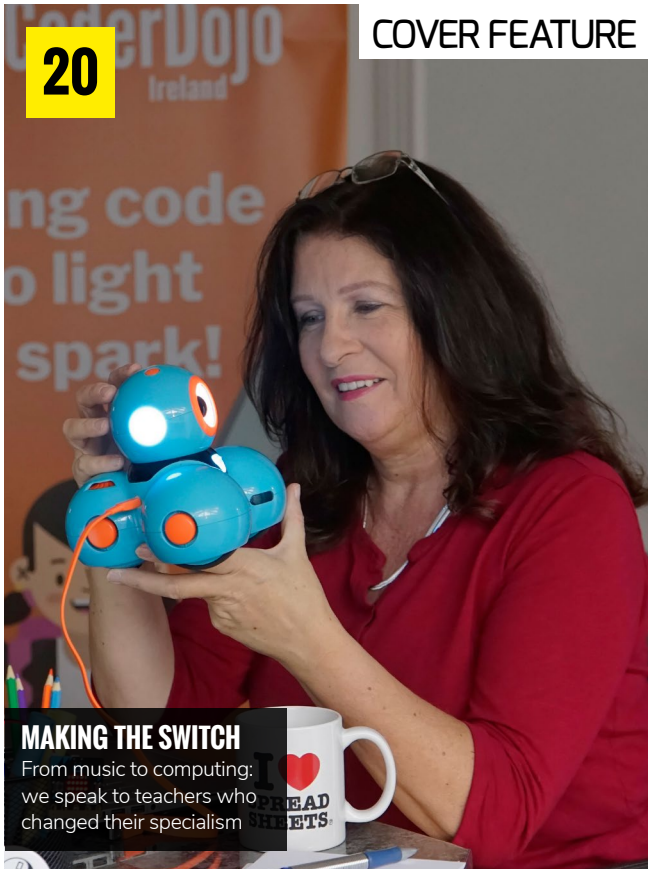
**Hello World is a joint collaboration:**

Raspberry Pi
bcs The Chartered Institute for IT Academy of Computing
CAS

This magazine is printed on paper sourced from sustainable forests and the printer operates an environmental management system which has been assessed as conforming to ISO 14001.

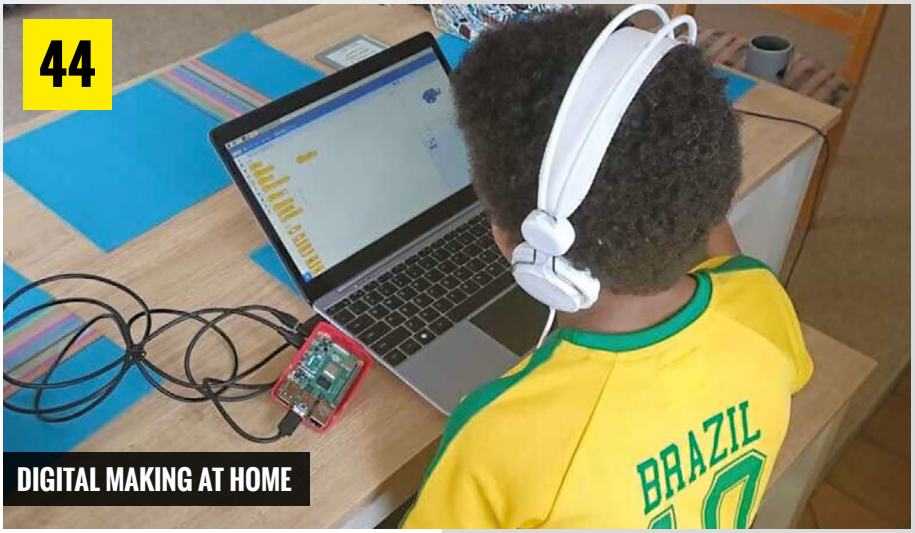
Hello World is published by the Raspberry Pi Foundation, 37 Hills Road, Cambridge, CB2 1NT. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to skills, products, or services referred to in the magazine.

# (HW) CONTENTS

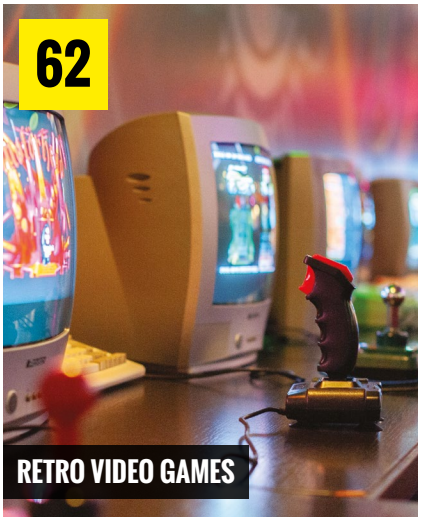





## NEWS, FEATURES, AND OPINION






Safety reminders in the classroom

# THE NEW NORMAL: TEACHING PROGRAMMING REMOTELY

School shutdowns and social distancing measures have required teachers to make quick adaptations to the way programming is taught

Sue Sentance

**With many schools across the world currently shut due to the coronavirus pandemic, teachers are having to adapt rapidly the way they teach students to code. The disruption caused by intermittent school closures means it is difficult for students to maintain the continual practice they need in order to learn programming concepts. However, teachers' sheer determination to provide their students with a quality education experience at home means that some benefits are emerging through the effective use of technology.**

During school shutdowns, such as the current situation in the UK, secondary teachers have quickly adopted online programming environments such as Repl.it, Trinket, and .NET Fiddle. One teacher explained how the platforms allow teachers and students to share code online: "The first part [of code] at the top might have been an example, so they would then modify that … Then towards the bottom of the page, they'd have an opportunity to make their own and to do their own creation of code."

However, access to devices and sufficient internet connection remain problems for many students, despite attempts from governments and charitable organisations to provide computers to students who need them. One teacher explained: "We started to use Repl.it, but what we found, with where the school is, the availability of devices that can handle doing two things at once, or even doing something using Repl.it at all on a mobile device — it just doesn't work. So we ended up just doing theory online that can be done on a bit of paper."

Where programming online doesn't work well, some teachers report that they focus on written exercises to support their programming teaching and formatively assess progress. Some students are using exercise books and worksheets at home. Alternatively, a number of teachers use environments such as Microsoft Teams or Google Classroom to get students to take screenshots of their code, annotate it, and share with their teacher. Other teachers are choosing to focus on theory that isn't related to programming.

## Programming in school also sees a change

When schools in the UK were open in the autumn, safety measures meant that teachers had to adapt the way they taught students to code. They divided students into bubbles, with changes made to timetabled rooms. In most schools, teachers could not be close to individual students in the classroom to have quiet conversations.


Rooms have been adapted for distancing


Safety measures to make schools safer

The age-old practice that programming teachers use of looking over shoulders, to spot errors and support students who may be stuck and frustrated, was no longer possible. One teacher reported at the time: "We are restricted in our movements. I can have a conversation, but it tends to be quite loud. Everybody has to sit quite quietly if somebody needs help … Otherwise you're shouting across the classroom because you can't move to see them."

Another teacher reported that she was not allowed to let students talk to each other at all. So where she might previously have asked students to work through a program together, she instead encouraged them to use the rubber duck technique — where a student explicitly explains the purpose of each line of code — to debug their programs by themselves. Computer science education research has shown that working together to share code, and classroom talk about programming — a chance for students to verbalise the way code works and practise coding vocabulary — are invaluable approaches to learning the subject. Teachers and students have very limited access to these methods while following coronavirus safety guidelines in schools.

> "RESEARCH SHOWS THAT WORKING TOGETHER TO SHARE CODE IS AN INVALUABLE APPROACH TO LEARNING

Several teachers reported using software on a desktop computer at the front of the class to see all their students' screens at once and monitor their programming progress. One teacher described being able to see one screen "go red" (with a Python syntax error) and knowing instantly which student needed help: "Rather than saying 'Where are you stuck?' and them trying to explain to you, I get it from [the software] and I can see how much they've understood."

## "YOU WANT TO BE ABLE TO WALK OVER …"

"The Atlanta Public Schools that I work with were shut down until the teachers went back this Monday [1 February]. They have not been in school since March of last year, and so teachers went back and students have the option of either coming back or staying virtual.

"When it first started, the major scramble was getting devices out to kids. So they had to get devices; get hotspots. Teachers pivoted online quickly. Now, as the kids come back, the teacher is doing Zoom classes, but she has anywhere from six to 15 kids sitting in the class, on their computers as well as the teacher teaching.

"The lack of proximity to the kids - everyone I've talked to says that it's one of the hardest things, because you want to be able to walk over… Most teachers that are in the business like kids. And so they want to go and be helpful. And I think that's one of the hardest parts of this, that they can't reach out and be like, 'OK, let me show you this.'

"The one good thing about computer science is a lot of the curricula were already online … they were doing a lot of [the interactive online learning platform] codeHS at the high-school level. Where we found we had issues is with physical computing. So, whereas we would go out, give kids drones, Raspberry Pis, and micro:bits, we don't have enough to give every child one. Georgia Tech purchased some for our teachers, but I don't know how other people are handling that, because the financial situations here vary; we have schools where you have kids that might have a lot of money, through to foster kids and homeless kids [who may have less or none]. So physical computing, I think, has kind of been harder."

- **Yolanda Payne**, Georgia Tech Constellations Center for Equity in Computing

## "TEACHING REMOTELY HAS ENABLED ME TO TEACH THEM BETTER"

"If you compare teaching live online now, to teaching in a classroom in September, where we couldn't get near a child, we had to stand in a box in the front of the room, with a mask on, and with all the windows open - it got very cold ... it was hard to see students' work. Now, we use OneNote, and I can see them typing, and I can give them feedback, which I could not do [when schools were open to students] in September. So in some ways teaching remotely has enabled me to teach them better.

"From September to Christmas, we could stand at the front of the class, and with a mask on, I could go near a child for under a minute, but still two metres away. What I did find useful was software called Impero ... you can take over a child's screen and look at what they're doing. I know other schools have used it for behaviour management, but we've always shied away from that, because we felt like you should behave because you're being told to behave, not because I can control your computer. But in the September to December bit, it was actually really useful.

"Normally, when you're trying to help a child to debug their code, you'd stand quite close to them, and look at their computer, and sometimes you take over and type, or say 'Stand up and let me sit down and help you.' We obviously could not do that, which was very frustrating, but we could use that technology to take the machine over, and get them to stand two metres away from us, and talk through their code, which was useful. But now, we can do that in real time with sharing screens and what have you. With [Microsoft] Teams meetings, if you make a child the presenter in the meeting, you can request control of their computer."

- **Katie Vanderpere-Brown**, Saffron Walden County High School in Essex

This is a significant change in pedagogy, as teachers might previously have encouraged students to work through a programming problem to try to explain the cause and location of the error verbally. Dialogue such as this is important in a number of ways, from using correct programming terms, to being able to trace through a program orally. The restrictions may mean that teachers just need to tell the student where the error is, which is much less effective in the longer term. Where students need to be socially distant, some teachers reported that they used breakout rooms within video conferencing environments, rather than physically close groups, so that students could work on tasks and also confer with each other.

> "IF STUDENTS NEED TO SOCIALLY DISTANCE, USING VIDEO CONFERENCING BREAKOUT ROOMS MEANS THEY CAN STILL CONFER

### Becoming independent

Despite the many difficulties, some teachers said that their students had become more independent, particularly the older ones: "I would say they're more independent, which is fantastic. Especially the Year 10s seem to be working harder, because they know I can see what they're doing all the time as well."

When schools were open, there were many cases of teachers either having coronavirus or needing to self-isolate. Some teachers had success with teaching their classes from home while their students were in the classroom. One teacher explains: "I had to isolate for two weeks, and I taught from home into school ... It was really successful. And we used the video conferencing chat as a 'back channel' to ask for help. So I wouldn't let the children come on that for anything else apart from asking me for hints."

This teacher explained that a surprising benefit was that the learners could see the explanations that were given to classmates, which they wouldn't normally be privy to. Even so, another teacher who taught in this fashion said that her students were pleased when she returned in person: "One of the students turned around and said 'Miss, I'm so pleased you're back. This programming lark is too difficult when you're not in the room.' Even though I can't physically go near her, it's the reassurance of having me there and the instant questioning, I think." (HW)

Homeworking has made some students more independent

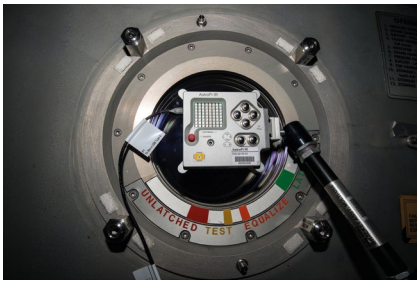

ESA astronaut Thomas Pesquet with the Astro Pi computers onboard the ISS

# ASTRO PI: MISSION ZERO

**Ross O'Neill** shares details of the project in which students can send code to the ISS

Ross O'Neill

**The popular Astro Pi Mission Zero initiative returns, to recreate the opportunity for young people in ESA Member States to write a few lines of code that will run onboard the International Space Station (ISS).**




The Mission Zero initiative is designed for beginners and young participants up to age 14

The code they write will be sent into space and run for up to 30 seconds on the two Astro Pi computers that are onboard the ISS. This year's challenge involves using a different sensor than in previous years: we are inviting young people to measure the humidity (rather than the temperature) in the Columbus module, which is the European-built ISS module where the Astro Pi computers live.

In recognition of the changes brought about by lockdowns and school closures, we have changed the entry criteria so that young people coding alone can take part as well, rather than only teams of participants.

Mission Zero is designed for beginners and young participants up to the age of 14. It can be completed online in around an hour by following our step-by-step project guide. Taking part doesn't require any previous coding experience or specific hardware — all young people need to get involved is a web browser!

Every participant from an ESA Member State will receive a certificate showing exactly where the ISS was above the Earth while their program ran.

Get your learners involved today! More information about taking part in Astro Pi Mission Zero, including the list of eligible countries, can be found on the Astro Pi website, **astro-pi.org/mission-zero**. The deadline for participants in this year's challenge is **19 March 2021**. (HW)

# SUPPORT FOR TEACHERS WITH HOME LEARNING

The National Centre for Computing Education in England has free resources that can be accessed by teachers globally

Victoria Temple

**With schools around the world adapting to different types of teaching and learning in response to the global pandemic, the National Centre for Computing Education in England (NCCE) has been working to support teachers and learners with video lessons and online community meetings.**

Dave Gibbs, STEM Learning's senior computing and technology specialist, and part of the National Centre's team, praised teachers in England, who he says have "performed minor miracles to keep children learning". He continued, "Schools are now past the emergency response phase of remote teaching and into more regular patterns of provision."

## A comprehensive set of computing materials

Support from the NCCE has been designed with both classroom and remote teaching in mind. "One of our biggest projects for computing teachers that we've worked on over the past two years is the Teach Computing Curriculum, a comprehensive set of free computing classroom materials for Key Stages 1 to 4 (learners aged 5 to 16)," says Carrie Anne Philbin MBE, director of educator support at the Raspberry Pi Foundation, who leads curriculum development for the NCCE. The materials comprise lesson plans, homework, progression mapping, and assessment materials. They were created as part of the NCCE, but they are freely available for educators all over the world to download and use at **helloworld.cc/tcc**.

"Working with Oak National Academy, we've turned the materials from our Teach Computing Curriculum into more than 300 free, curriculum-mapped video lessons for remote learning," she continued. "They are freely available for parents, educators, and learners to continue learning computing at home, wherever they are in the world." In December, over 150,000 computing lessons had been started.

> "TEACHERS HAVE PERFORMED MINOR MIRACLES TO KEEP CHILDREN LEARNING

## Community support

To help with curriculum implementation, and new skills such as remote delivery, teams of NCCE subject matter experts and Computing at School (CAS) Communities have been supporting schools and teachers through online training, meetings, and knowledge sharing.

Chris Hillidge, who leads the NCCE Computing Hub for Merseyside and Warrington in England, said: "The first lockdown accelerated our use of online resources and remote capabilities and, since September, we have seen some amazing innovative practice from teachers of all subjects. CAS has been a really useful support network, and back in May and June we saw increased numbers of teachers accessing the NCCE's remote CPD courses."

The teacher-led network CAS has seen increased attendance at its online meetings and events to share resources and ideas. Wendy Piccinini, one of their team of outreach managers, said: "Teachers have always turned to teachers to find out what works in the classroom, and they're doing that with remote delivery, too … Lunchtime sessions seemed to work well, fitting in with busy schedules."

Thanks to the CAS Communities, teachers are developing their practice in remote teaching. Chris Hillidge said: "The pandemic has definitely accelerated teachers' tech skills … it showed teachers that remote learning can be as simple or complex as you want to make it, so long as it is effective and learning takes place."

To access the full set of learning materials and professional development training, visit the National Centre for Computing Education website at **teachcomputing.org**. For further details of CAS Communities see **computingatschool.org.uk**. (HW)

# COOLEST PROJECTS RETURNS AS AN ONLINE SHOWCASE

Through Coolest Projects, young people are empowered to show the world something they're making with tech

Katie Gouskos

**Coolest Projects is the world-leading technology fair for young people. In 2020, more than 700 young people from 39 countries, including Ireland, Australia, Palestine, UK, USA, India, and Indonesia, shared their tech creations in the Coolest Projects free online showcase. The Coolest Projects team is running the fair online again this year, so that young people can participate safely, and from wherever they are located.**

## Creating projects for social good

Creator Abhiy began coding at age seven with Scratch and Kodu, taught in his local Code Club, and in 2019 attended Coolest Projects International in Dublin. He created a Climate Change project which he built in the block-based visual programming language, Scratch. Abhiy says:

"This was my first remarkable success in coding … which gave me more confidence and motivation for exploring coding even more."

In 2020, Abhiy wanted to create something with physical computing, and had the idea of making a home alarm system after seeing a rise in home robberies. Burglar Buster was part of the 2020 Coolest Projects online showcase and is a house alarm system made with a Raspberry Pi 4 coded in Python 3, and uses buzzers, sensors, LEDs, and a camera. Abhiy explains:

"So many burglaries going on in my local area inspired me to work on a Raspberry Pi–powered home alarm system coded in Python. The project was Tim Peake's favourite in the Advanced Programming category at Coolest Projects 2020. This encouraged me even more to learn further."


A young creator at the Coolest Projects technology fair

## Taking part in Coolest Projects

Coolest Projects is a powerful motivator for young people and encourages them to develop skills in idea generation, project design and planning, coding and technology, user testing and iteration, and presentation. As with Abhiy, it also helps to build their confidence, and it's an opportunity to get creative, have fun, be inspired by their peers, and be a part of something truly special.

It is completely free to take part in, and anyone, anywhere in the world, will be able to view the submitted projects on the Coolest Projects online gallery. Young people can participate with whatever they're making.

If you know a young tech creator, encourage them to submit a project, whether it's an animation, website, game, app, robot, or anything else they've built with technology. All project types at all levels of skill are encouraged, from beginner to advanced, and it doesn't matter whether the project is a work in progress, a prototype, or a finished product.

Young people up to the age of 18 must register a project at **coolestprojects.org** before 3 May 2021 to be part of this year's Coolest Projects showcase. (HW)

# GOING UNPLUGGED

**Catherine Elliott** shares ways to teach key computing concepts to learners with special educational needs and disabilities through unplugged activities

**Unplugged activities occur away from the computer and can be used to teach abstract concepts in computing.** They are popular with teachers across primary and secondary schools and I have written in previous issues about the benefit of using them with learners with special educational needs and disabilities (SEND). In this column, I'll look at the benefits of the unplugged approach through the lens of the Universal Design for Learning Guidelines, and explore the use of semantic profiling to ensure the teaching of concepts is effective.

## Why use unplugged activities?

Many students with learning difficulties or sensory impairments struggle to make sense of abstract concepts. We can make the abstract more concrete through unplugged activities by using objects, movement, sound, and other physical representations. As professor of computer science Paul Curzon writes, "By providing a physical representation, the learner can point to and ask the question at the level of the analogy rather than having to fully verbalize it at the technical level."

The varied and sensory nature of unplugged activities also helps us include and engage more students by having multiple means of engagement, representation, action, and expression. These are the key elements of the Universal Design for Learning Guidelines (**helloworld.cc/udl-guidelines**), which were developed to help teachers create more inclusive lessons.

## Multiple means of engagement

Young people with SEND, particularly autistic students, often have very specific interests. We can harness these interests through an unplugged activity, for example, learning about algorithms using Lego bricks, or creating pixel art to investigate digital image representation. Activities can easily be made age-appropriate: students working below age expectations can learn about simple concepts within a context relevant to them, rather than relying on activities appropriate for younger children. This will help learners to sustain their efforts and concentration, and maximise learning.

## Multiple means of representation, action, and expression

We can also adapt activities to ensure accessibility for a range of learning and sensory difficulties. Visually impaired learners can learn about sequence by creating musical algorithms. Students with poor literacy can investigate complex concepts such as modelling repetition in algorithms through dance.

Using familiar contexts for an activity reduces cognitive load so that the learner can concentrate on the specific concept being taught. Technology itself can be a distraction, and an unplugged activity to introduce key concepts is often beneficial.

Catherine is the SEND lead for the Sheffield eLearning Service (sheffieldclc.net), and she has spent the last five years working on ways to make computing accessible for all learners. She is a member of the CAS Include working group, and co-leader of the Sheffield and South Yorkshire Secondary CAS Community (**@catherinelliott**).

## Semantic waves

The efficacy of unplugged activities for teaching abstract concepts is not always guaranteed. Make explicit links between the practical exercise and the concept in a computational context. We can teach pupils to give clear instructions to a robot to make a jam sandwich in an incredibly engaging way. However, without using the technical language (for example 'algorithm' or 'sequence') or the opportunity for children to make links with the abstract concept of 'algorithm', it's a literacy task about giving precise instructions.

## RESOURCES

- Barefoot Unplugged activities (filter for SEND-specific materials): **helloworld.cc/barefoot-resources**
- Inspiring classroom activities from Teaching London Computing: **helloworld.cc/tlc-unplugged**

One approach to ensure we teach unplugged activities effectively is using semantic waves and semantic profiling. Jane Waite et al. wrote about this in issue 10 of Hello World, and there is a Pedagogy Quick Read on the subject available at: **helloworld.cc/semantic-waves**. Semantic waves are part of Legitimation Code Theory, and enable us to analyse lesson activities in relation to two concepts: semantic gravity and semantic density.

Semantic gravity relates to the context in which a concept is presented — the more relevant and familiar the context to the learner, the higher the semantic gravity. Semantic density relates to the complexity of the meanings — so a task presented in everyday language, such as "Put the objects in order following the instructions", would have a weaker semantic density than one couched in technical, academic language, such as "Sequence the objects according to the algorithm."

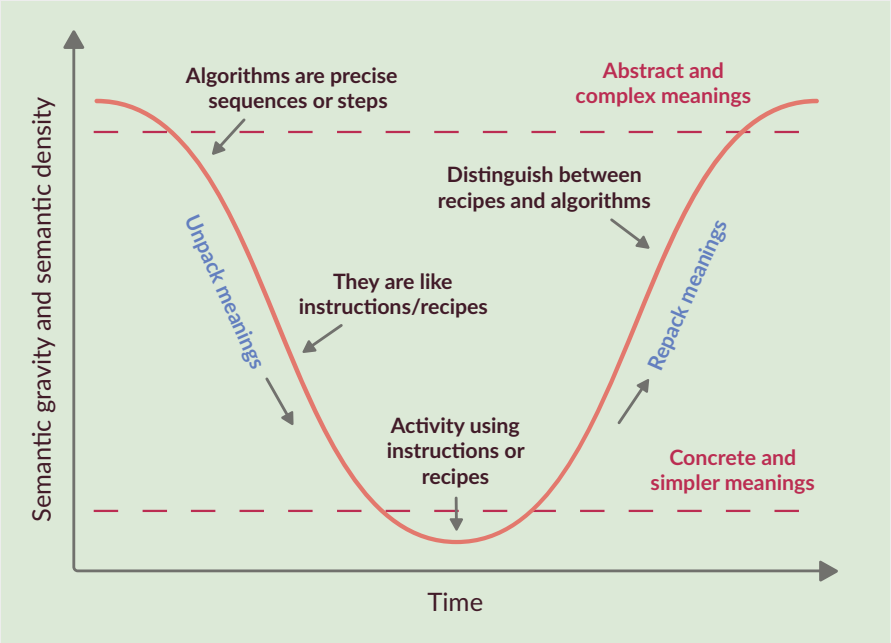
Our ultimate aim is for students to master a deep understanding of a concept and the technical language that accompanies it (high density/low gravity). An effective lesson builds this understanding by following a semantic wave: introducing an abstract concept, then unpacking the meaning using analogies or practical activities with familiar contexts and everyday language (low density/high gravity), then repacking knowledge so pupils can use technical language and describe the abstract concept separate to the context. Repacking may summarise the main points, asking learners to identify examples of the concept or explain what they've learnt.

I am interested in what the semantic profile looks like when teaching students with SEND — is it, or should it be, any different? Reflecting on my own experience, one of the challenges for students with SEND is generalising what they've learnt. They may benefit from several consecutive unplugged activities in which they revisit the concept several times using different contexts. For many students, full repacking may never happen, and this is where differentiation may occur. Undertake the same activity, and some students will explain the links between the activity and the abstract concept; others may never make the jump to the full abstract concept. So in the example of the Musical Sequences Activity on Barefoot (**helloworld.cc/musical**), some pupils will learn about algorithms and sequence, while others learn about the concept of controlling a computer. Students working well below age expectations may simply enjoy a lesson making music on a computer.

Next time you undertake an unplugged activity with a class, consider the semantic profile of your activities — how might you adapt the activity for different groups of learners and how do you repack the concept at the end of the lesson? When making links to the concepts, check to ensure your students have made these connections. (HW)

## FURTHER READING

Curzon, P. (2018). Teaching computing concepts. In: S. Sentance, E. Barendsen, C. Schulte, eds, *Computer Science Education: Perspectives on Teaching and Learning in School*, 1st ed. Bloomsbury, p. 95.





# ANNE-MARIE IMAFIDON

## "WE NEED TO MOVE THE SOCIAL NORM"

Anne-Marie Imafidon tells **Sian Williams Page** how she devised the approach of the social enterprise Stemettes, and her plans for wider social change in 2021

**In 2013, while working at Deutsche Bank, Anne-Marie Imafidon started Stemettes, a social enterprise with a mission to inspire the next generation of girls and non-binary people into STEM careers. Since then, the organisation has held events — each with the mantra 'free, fun, food' — for more than 45,000 young people. For her groundbreaking work, Imafidon was awarded an MBE in 2017 and an honorary degree from the University of Bristol in 2019. We spoke in December, at the end of another busy year for Imafidon: she had published her first book — *How to Be a Maths Whizz*, for young children — alongside leading Stemettes through a period of extraordinary change due to the pandemic.**

"We had to kind of put down the train track as we were steamrolling ahead … We have translated to allow that kind of 'free, fun, food' thing to still exist. We've been posting food out. We've been still keeping the fun up. We're still having music in the background, still allowing people to kind of connect, and build, and get technical."

Imafidon explains that the adaptations they've made have also considered young people without access to technology at home: "We have had to send laptops out, we have had to pay for data packages, we have sent dongles."

One upside of the way they have adapted to the pandemic, she explains, is that Stemettes can now reach young people outside of the UK. In their summer programme, nearly 400 young people gained certifications in subjects like Python and cybersecurity: "We've had

## RESOURCES AND FURTHER READING

Stemettes (**stemettes.org**)

The Everyday Sexism project (**helloworld.cc/sexism**)

CAS Include (**helloworld.cc/casinclude**)

Georgia Tech Constellations Center for Equity in Computing (**helloworld.cc/GTConstellations**)

Royal Academy of Engineering diversity resources (**helloworld.cc/RAEng**)

The National Centre for Computing Education's Gender Balance in Computing project (**helloworld.cc/GBIC**)

people from 13 or so other countries joining us for these programmes. And because everyone was virtual, they felt as if they literally were here, because everyone had the same experience."

Imafidon explains that she came up with the idea for Stemettes after attending an event for women working in technology in 2012: "I remember the feeling of being in a majority-female technical environment. Tech has always been my thing. But being there, I was like, 'How great is this? This is what it feels like to be in the majority in a technical space? Who knew?' It was like, 'Here is where the magic can happen.' And so it was taking that feeling and that experience, and mapping it to a slightly younger audience, and giving people that positive experience in their formative years."

The Stemettes' approach of fostering a sense of community was instinctive, she explains: "That's just what I'm like … I'm very much at the centre of whatever community I'm in and I like to enjoy time with my friends. I was never a gamer. So that kind of singular thing you see in the movies just isn't me. I've never watched *Star Trek* or any of that kind of stuff, and I don't play Worms. So I think for me it was, 'Look, I'm me, and I'm technical, so you don't have to be that geek thing … I went to Oxford. I did maths and computer science. I finished a year early, I kind of know my stuff. And so the whole *Star Trek* thing can't be a prerequisite. The whole 'Don't talk to people' thing can't be a prerequisite, because I did.'"

Imafidon explains that by focusing on girls and non-binary young people, it felt as if she was going against the tide of other organisations. "When we started, a lot of people were like, 'No, no, no, don't focus just on the girls, you can't do that.' And it's like, 'Why can't I? Why can't we be overt?' Because being subtle to whatever level hasn't worked and being serious and being curriculum-led and being so very focused on the technology above all else, it doesn't help. It's also not really the way the tech industry is set up now."

Stemettes measure their success by looking at where their alumni are working by age 25, rather than focusing on getting more girls to take computer science at school. "As much as I want to tell them to take it, lots of girls can't, because they don't have the option. And the girls that do have the option, a lot of them end up having an awful experience in that classroom. A lot of them, it's because of their peers and their teachers. Some of them, it's because of the curriculum … I don't want them to have to go through that.

"If you look at Laura Bates' Everyday Sexism project, the number of those that are computer science lesson experiences … Until we stamp out the blatant sexism and racism that we're seeing in these classrooms, I can't say at a Stemettes event, 'Everyone here must now choose computer science.' All I can say is, 'This person works at NASA, and here's what they did to get to where they are. So here's why, when you're sat in your computer science classroom and people are treating you how they're treating you, you might want to hold on to what you're studying, because it's not about your peers. It's about what you'll be able to do next.'"

While Imafidon thinks there should be a shift in the English computer science curriculum — "What you're going to have to do with computer science, you write non-white men into it and you write the altruistic end of it into it, and you write creativity at the core of it" — it's not her focus. Instead, she says, she's trying to cause a shift in the attitudes of people who have already been through the school system. She explains that she has plans for a second book project in 2021, this time for adults, alongside work with NATO, G7, and the Institute for the Future of Work, where she is a trustee.

"It needs to be a societal thing where all of us wake up … We have bulletproof vests, we have GPS, we have WiFi. All of these things we have because of women, but no one has that association …

"I've said since the beginning of Stemettes that we need to move the social norm. I think it's something we've been able to do for the fifty thousand-odd young people we've worked with … The next step for me, at least for next year, is definitely going to be, how do we go up a level?

"With the institutions at the top of society, how are we going to push things in a particular direction? How are we going to ensure that if you're an academic as a woman, you don't have an awful time? How do we make sure that if you're a Black woman, you don't have the most awful time? And how do we make sure that if you're using the technology — you know, technology is power, right? — how do you know you're not abusing your power and creating more problems?" (HW)

***How to Be a Maths Whizz*** by Anne-Marie Imafidon, published by DK Children, is available now: **helloworld.cc/mathswhizz**.

#INSIGHTS

# UNIVERSAL DESIGN FOR LEARNING IN COMPUTING

STORY BY Hayley Leonard

**Universal Design for Learning (UDL) is a framework for considering how tools and resources can be used to reduce barriers and support all learners. Based on findings from neuroscience, it has been developed over the last 30 years by the Center for Applied Special Technology (CAST), a nonprofit education research and development organisation based in the US. UDL is currently used across the globe, with research showing it can be an efficient approach for designing flexible learning environments and accessible content.**

Engaging a wider range of learners is an important issue in computer science, which is often not chosen as an optional subject by girls and those from some minority ethnic groups. Researchers at the Creative Technology Research Lab in the US have been investigating how UDL principles can be applied to computer science, to improve learning and engagement for all students. They have adapted the UDL guidelines to a computer science education context and begun to explore how teachers use the framework in their own practice. The hope is that understanding and adapting how the subject is taught could help to increase the representation of all groups in computing.

## A scientific approach

The UDL framework is based on neuroscientific evidence which highlights how different areas or networks in the brain work together to process information during learning. Importantly, there is variation across individuals in how each of these networks functions and how they interact with each other. This means that a traditional approach to teaching, in which a main task is differentiated for certain students with special educational needs, may miss out on the variation in learning between all students across different tasks.

The UDL guidelines highlight different opportunities to take learner differences into account when planning lessons. The framework is structured according to three main principles, which are directly related to three networks in the brain that play a central role in learning. It encourages educators to plan multiple, flexible methods of *engagement* in learning (affective networks), *representation* of the teaching materials (recognition networks), and opportunities for *action and expression* of what has been learnt (strategic networks).

The three principles of UDL are each expanded into guidelines and checkpoints that allow educators to identify the different methods of engagement, representation, and expression to be used in a particular lesson. Each principle is also broken down into activities that allow learners to *access* the learning goals, remain engaged and *build* on their learning, and begin to *internalise* the approaches to learning so that they are empowered for the future.

## EXAMPLES OF UDL GUIDELINES FOR COMPUTER SCIENCE EDUCATION FROM THE CREATIVE TECHNOLOGY RESEARCH LAB

| Multiple means of **engagement** | Multiple means of **representation** | Multiple means of **action and expression** |
|---|---|---|
| **Provide options for recruiting interests**<br>Give students choice (software, project, topic)<br>Allow students to make projects relevant to culture and age | **Provide options for perception**<br>Model computing through physical representations as well as through interactive whiteboard/videos etc.<br>Select coding apps and websites that allow adjustment of visual settings (e.g. font size/contrast) and that are compatible with screen readers | **Provide options for physical action**<br>Include CS unplugged activities that show physical relationships of abstract computing concepts<br>Use assistive technology, including a larger or smaller mouse or touchscreen devices |
| **Provide options for sustaining effort and persistence**<br>Utilise pair programming and group work with clearly defined roles<br>Discuss the integral role of perseverance and problem-solving in computer science | **Provide options for language, mathematical expressions, and symbols**<br>Teach and review computing vocabulary (e.g. code, animations, algorithms)<br>Provide reference sheets with images of blocks, or with common syntax when using text | **Provide options for expression and communication**<br>Provide sentence starters or checklists for communicating in order to collaborate, give feedback, and explain work<br>Provide options that include starter code |
| **Provide options for self-regulation**<br>Break up coding activities with opportunities for reflection, such as 'turn and talk' or written questions<br>Model different strategies for dealing with frustration appropriately | **Provide options for comprehension**<br>Encourage students to ask questions as comprehension checkpoints<br>Use relevant analogies and make cross-curricular connections explicit | **Provide options for executive function**<br>Embed prompts to stop and plan, test, or debug throughout a lesson or project<br>Demonstrate debugging with think-alouds |

Each principle of the UDL framework is associated with three areas of activity which may be considered when planning lessons or units of work. It will not be the case that each area of activity should be covered in every lesson, and some may prove more important in particular contexts than others. The full table and explanation can be found at *Israel, M., Lash, T., & Jeong, G. (2017). Utilizing the Universal Design for Learning Framework in K-12 Computer Science Education. Project TACTIC: Teaching All Computational Thinking Through Inclusion and Collaboration. Retrieved from University of Illinois, Creative Technology Research Lab website: ctrl.education.ufl.edu/projects/tactic.*

## Applying UDL to computer science education

While an advantage of UDL is that the principles can be applied across different subjects, it is important to think carefully about what activities to address these principles could look like in the case of computer science.

Researchers at the Creative Technology Research Lab, led by Maya Israel, have identified key activities, some of which are presented in the table on the previous page. These guidelines will help educators anticipate potential barriers to learning and plan activities that can overcome them, or adapt activities from those in existing schemes of work, to help engage the widest possible range of students in the lesson.

## UDL in the classroom

As well as suggesting approaches to applying UDL to computer science education, the research team at the Creative Technology Research Lab has also investigated how teachers are using UDL in practice. Israel and colleagues worked with four novice computer science teachers in US elementary schools to train them in the use of UDL and understand how they applied the framework in their teaching.

The research found that the teachers were most likely to include in their teaching multiple means of engagement, followed by multiple methods of representation. For example, they all offered choice in their students' activities and provided materials in different formats (such as oral and visual presentations and demonstrations). They were less likely to provide multiple means of action and expression, and mainly addressed this principle through supporting students in planning work and checking their progress against their goals.

> "THE GUIDELINES HELP EDUCATORS ANTICIPATE BARRIERS TO LEARNING AND PLAN ACTIVITIES TO OVERCOME THEM

Although the study included only four teachers, it highlighted the flexibility of the UDL approach in catering for different needs within variable teaching contexts. More research will be needed in future, with larger samples, to understand how successful the approach is in helping a wide range of students to achieve good learning outcomes.

## Find out more about using UDL

There are numerous resources designed to help teachers learn more about the UDL framework and how to apply it to teaching computing. The CAST website (**helloworld.cc/cast**) includes an explainer video and the detailed UDL guidelines. The Creative Technology Research Lab website has computing-specific ideas and lesson plans using UDL (**helloworld.cc/udl**).

Maya Israel will be presenting her research at one of the online Raspberry Pi Foundation computing education research seminars on 20 April 2021. The seminars are free to attend and open to anyone from anywhere around the world. For more information about the spring seminar series, which focuses on diversity and inclusion, and to sign up to attend, please visit **helloworld.cc/RPFseminars**. (HW)

## FURTHER READING


- Israel, M., Jeong, G., Ray, M., & Lash, T. (2020). Teaching Elementary Computer Science Through Universal Design for Learning. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, pp. 1220-1226. dl.acm.org/doi/abs/10.1145/3328778.3366823
- Rose, D. H. & Strangman, N. (2007). Universal design for learning: Meeting the challenge of individual learning differences through a neurocognitive perspective. *Universal Access in the Information Society*, 5(4), pp. 381-391. dl.acm.org/doi/abs/10.1007/s10209-006-0062-8

# "IT WAS A JUMP IN WITH BOTH FEET"

**Sian Williams Page** and **Amy O'Meara** spoke to five educators to find out how – and why – they switched subject specialisms to computing

Many teachers working now didn't have the opportunity to study computing when they were at school, and weren't aware of it as a subject at university. We wanted to hear what it's like to transfer into teaching computing from another subject area, and why some teachers make that choice. We spoke to schoolteachers and volunteers in extracurricular clubs in the UK and Ireland, and heard a diverse range of stories. Some teachers had loved tinkering with computers since they were young, but didn't have the opportunity to study computing themselves. Others picked up an interest only after covering computing lessons because of timetabling constraints. We heard about numerous challenges — of learning to program alongside students, and of surprising changes to some students' behaviour when switching from a PE field to a computer room, to name just two — but we also heard about the joys that teachers found in applying their skills to a new field, and in helping their students create things with technology.

## "A level computer science is an incredibly wide-ranging subject"

**Simon Baldwin, Abbeygate Sixth Form College, Bury St Edmunds, and Landmark International School, Cambridge**

When I was a kid, I had an 8-bit computer in my bedroom and learnt how to code, but computer science wasn't an option at my school, so I didn't really know how to pursue it as a subject. I did a PGCE straight after my physics degree, and then went straight into science teaching; I was a physics teacher for 20 years. Then, in 2013, I started to think I should really be doing what I enjoyed most, and I was reading magazines about computing rather than physics, so that was more of a passion for me. We didn't have a computing teacher at the school where I was working, so I asked them if I could retrain.

My school was really supportive, so I did a software programming course at the local technical college, which was in C#, and we bought a Raspberry Pi, initially just for me. And I also went on some Python teacher training courses. Within a year, we had a class set of Raspberry Pis and the computers in the lab were running Linux. In 2019, I became a National Centre for Computing Education (NCCE) associate facilitator for computing.

This year I'm teaching A level computer

Simon Baldwin enjoys learning alongside his A level students

science, and it's an incredibly wide-ranging subject: bits of it are like science, bits of it are like maths, and bits of it are like social sciences as well. We do a lot about the implications of computer science, about how it's changing the world, and the ethics of it.

I particularly enjoy teaching the bits that maybe I'm not as familiar with, where I'm learning with the students. Recently, I've been doing a project with my A level students on HTML, CSS, and JavaScript. You become more of a guide, more of someone alongside the student, and you're teaching students how to learn as much as just distributing knowledge.

I actually think in some ways it's a healthier way to teach, because you're still directing, but it's fresher and it's more fun. You just have to celebrate the students who know more than you about a particular area.

## "Mentoring is more than just knowing what the correct answer is"

**Sandra Maguire, champion of CoderDojo Dún Laoghaire and Coolest Projects International team member**

After I left school, I worked at Ulster Bank, and then I decided I wanted to travel somewhere. So I managed to get myself a job in marketing in Wall Street, New York. While I was there, I studied marketing at New York University, and then came back here, and ended up in stockbroking.

Then my kids came along, and my husband works in film, so it was really challenging to keep two heavy jobs going. So for a long time I was at home with the kids, and doing part-time work. I was always into computers, so I would do things like accounts for people, or design newsletters; anything they would pay me for. I was volunteering at my son's school and the principal asked if I was interested in covering for the school's secretary, so I went for it, and I ended up staying there for a year. And I kind of liked the whole education thing — and while I was there, I computerised everything in sight!


Sandra Maguire founded the Dún Laoghaire CoderDojo

> YOU JUST HAVE TO CELEBRATE THE STUDENTS WHO KNOW MORE THAN YOU

Around that time, I learnt that it was very difficult to get programmers here in Ireland. And if you could get them, they cost an absolute arm and leg. In Ireland in 2012, we had about 15 percent unemployment, but we were importing programmers and tech specialists. I have no problem with people from other countries coming to Ireland, but I thought it was a shame that kids growing up in Ireland didn't have the skills that we were crying out for. In early 2012, I went to a Web Summit talk and I heard James Whelton talk about CoderDojo. And I just thought there should be a CoderDojo in Dún Laoghaire. So that's kind of where my whole background brought me to — thinking that learning to code would be really good for young people.

When we started, the mentors at the club were spending a lot of time helping the children with Scratch, but they wanted to be able to show more advanced stuff. So we offered to teach the parents Scratch, so that then the parents could be Scratch mentors. We started the normal sessions a week later in the next term, so that we had a session just for parents instead, which I joined in on, getting a basic training course in Scratch. I've learned more since from watching mentors show how to use various types of technology.

Now, the kids might ask me something and I'll be like "Ah! They're asking me something!" and I'll be thrilled with myself if I can answer it. But I also learnt that mentoring is more than just knowing what the correct answer is. I am able to help them learn how to find a solution, and I can teach them how to help each other.

The CoderDojo at Dún Laoghaire, where Sandra Maguire supports young people to learn to code

## "I had so many ideas, I said OK immediately"

**Richard Amponsah, St James' CE Primary School in Bermondsey**

This is my fourth year at my primary school, and my first year as computing lead. I have a background in sports coaching and special needs.

Before teaching, I was doing community work for Millwall Football Club as a teaching assistant in a school for children with autism, and spent three years as a sports mentor at a pupil referral unit.

We have an IT coordinator at my school, who deals with making sure the tablets are all sorted, the WiFi is working smoothly around the school, and other important aspects to ensure that technology is working efficiently. There were some days where she asked my class to try out something new, and I showed an interest. And then, last summer, the head teacher called me and said, "I've seen that you've been quite interested in IT. Would you want to be our computer lead?" I said OK immediately – I had so many ideas.

Richard Amponsah supports other teachers at his primary school with implementing the computing curriculum

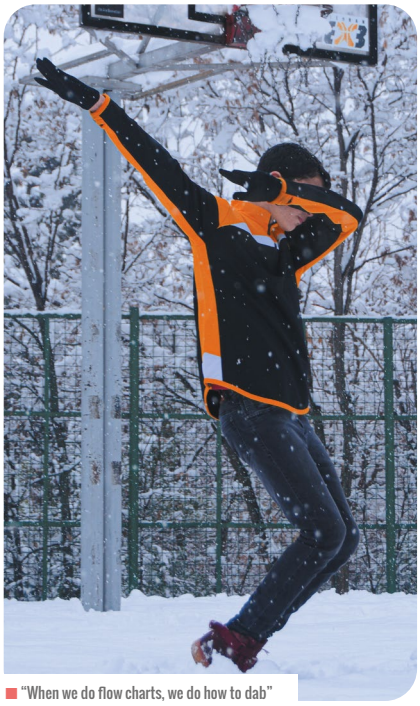
> "CPD COURSES SHOWED WAYS TO MAKE COMPUTING INTERACTIVE AND ENGAGING

Over the holidays, I took a couple of Raspberry Pi courses: one about computing with SEND, which was amazing, and one that included a lot of Scratch as well. I also took about six different STEM courses, and I use Teach Computing, and Barefoot for resources. I have decent IT knowledge, but I didn't have curriculum knowledge, so it was really good over the summer holiday to go and get those CPD courses in. They showed various ways of making computing more interactive and engaging, and not stressful or strenuous. You can actually teach computing without exclusively saying, oh, this is a computing lesson.

In this generation, a lot of children are good with tablets, and games, and coding, but they're lacking Microsoft Office skills, like Word, PowerPoint, and Excel. When you get to secondary, you tend to move away from an iPad to a computer, and there are just so many differences, like the shift button on the computer might be different to an iPad. The Caps Lock button might be different on a computer compared to an iPad. So there are so many things that we need to start teaching them.

Getting staff on board and excited about computing will ensure the children are excited too. I was pretty excited to get started and share some ideas. I'm looking forward to what the computing future holds.

Helen Brant went from teaching music to computing



"When we do flow charts, we do how to dab"

## "Coding is another language, music is another language"

**Helen Brant, Co-op Academy Priesthorpe**

My degree was in electronic music, but it was right at the start, when we were still using 3.5-inch floppy disks and DAT recorders — no one had an iPhone or GarageBand. Eleven years ago, I moved to my current school to be head of music, and I taught a lot of music technology. We had computers that could run music software like Logic Pro and Sibelius.

I took on an associate senior leadership position in ICT, and then, when the school was really struggling to find a computer science teacher, I started teaching computer science as well as music. When the head of computer science left, I was in the playground and I jokingly said to the assistant principal, "Oh, maybe I should just go be head of computer science for a bit", as I wanted a new challenge. She said, "Would you do it?" and I was like, "Oh? OK!" I took a temporary contract for the role, and I switched to teaching computer science and media.

I did the NCCE Computer Science Accelerator course. My school allowed me to go to the face-to-face and online components, and by the summer, I had done the exams. Last year I had to reapply for my job, because it was only temporary, and I've got it permanently now.

Music is another language, just like coding

Teaching music is much more me jumping up and down at the front of the classroom, and computer science is much more, "Sit down and get on with your work", but there are definitely similarities. Learning to program: it takes practice, and that's the same as learning a musical instrument. Coding is another language, music is another language: you can't just do it one day and then you'll be OK; you've got to do it daily.

I've spent so much time being creative in teaching, because in music you've got to be creative, and it was quite nice to be doing that with unplugged computer science lessons. The computer science teachers I've worked with are very much of the mindset that we sit down and get on with work, whereas I'm like, "No! We can do it like this, we can get sweets, we can do networking by getting them to stand in a line, we can get Lego out." I was doing a lesson on logic gates, and we did it as a dating app. When we do flow charts, we do how to dab,

Teachers can switch from the field to the classroom

or how to do the Fortnite dances — the students do a flow chart, and program it, and then get someone else to follow the flow chart, spotting why it doesn't work. When we're doing the different schemes of work, we try to make it more fun: playing with micro:bits, making Tamagotchis, and it's more about them taking ownership and experimenting.

## "I've gone from shorts to shirts"

**Richard Bateman, Alderman White School, Nottingham**

I went into teaching PE straight from school. So I've never left school, and I taught PE for over 20 years. Then, about five years ago, I started picking up some computing on my timetable, because there were too many PE teachers and not enough computer science teachers. I really enjoyed being in a classroom for a change, after so many years on a field. I was looking at it, thinking, "It's getting colder and wetter outside; I need to go do something else." And then this opportunity came up to go in as head of department. Now I've gone from shorts to shirts.

When I started teaching computing, it was a jump in with both feet. When I started picking up some lessons, it was teaching by PowerPoint — with the idea that students could get on with it, with a teacher just watching. But I wanted to be a bit more into it. So I learnt coding with them — that started my interest in computer science. More recently, I've done a few Computing at School courses, and I did the NCCE Computer Science Accelerator course during lockdown.

I'm quite prepared to learn from the pupils. Sometimes I've got them to share their screen with everybody else and they've taken over the teaching role. The PE style of teaching was like, "Can you do this? Right, show everybody else", so I've taken that into computer science as well.

There are so many resources available to help teach computing, it's almost like being in a candy store. And that's a great thing, but I've found myself picking lots of things. So I need to focus on picking what works well – it's a balancing act.

I certainly found that some of the pupils who were really engaged in PE, if I sat them in a computer science classroom, they were completely different, and sometimes they were the worst-behaved. And I found it was different teaching mixed groups; in PE I was mainly teaching an all-boys group or an all-girls group, not mixed groups, and the dynamics change a lot.

Richard Bateman spent 20 years teaching PE before becoming head of computing at his school

If anybody else is thinking of changing to computing, my advice would be that if you've got some free time, try and get into people's lessons and gauge your interest. That's how it happened to me: I was made to teach it and it was like, "Oh, I like doing this." (HW)

**DAVE GIBBS** STEM LEARNING

# TEACHING COMPUTING: WHAT IS A SPECIALIST ANYWAY?

Continued professional development offers a pathway into teaching computing for educators from diverse subject backgrounds — and computing is all the richer for it

**When asked to think back to their school days and computing experience, that memory will be patchy for many people.** I remember driving a taxi around a map of a town every week, while our teacher looked on, both mystified and terrified, by the new, beige machines in the school's only IT suite. For most of us, the depth of learnt experience in computing is unlikely to compare with our experience of established subjects such as maths or English.

Access to computer studies O level was haphazard back in 1985; at its peak, there were around 60,000 candidates per year in England. As schools switched to ICT, student numbers fell drastically, and by 2013 computer studies had almost disappeared off the radar. It is not surprising, then, that today's computing teacher workforce has the highest proportion of non-specialists among the English Baccalaureate (EBacc) subjects. But what is a specialist computing teacher anyway? The most restrictive definition is someone who holds a degree in the subject they teach. However, a student in the UK is more likely to be taught computing by someone with a degree in business studies than computer science. Science and maths teachers teach a significant proportion of computing lessons, too, bringing contexts and teaching approaches from across the curriculum that enrich this newly rebirthed subject. Many will have undertaken CPD to gain an understanding of the subject or to enhance existing knowledge.

A 2016 report by the UK's Department for Education looked at the impact of specialist and non-specialist teaching on pupil outcomes in England, and found little evidence that a teacher's qualifications significantly impacted outcomes.

Of course, that doesn't mean subject knowledge isn't important. In 2013, researchers in the US asked a group of science teachers to sit a multiple-choice test originally written for students. Unsurprisingly, the results found that the teachers who did better on the test were more effective at their jobs. Even more interesting is that the teachers who could identify common mistakes students would make were also more effective. So it's a broad but thorough knowledge of the teaching content, and of how students understand and learn that content, that matters.

## Easing the transition into teaching computing

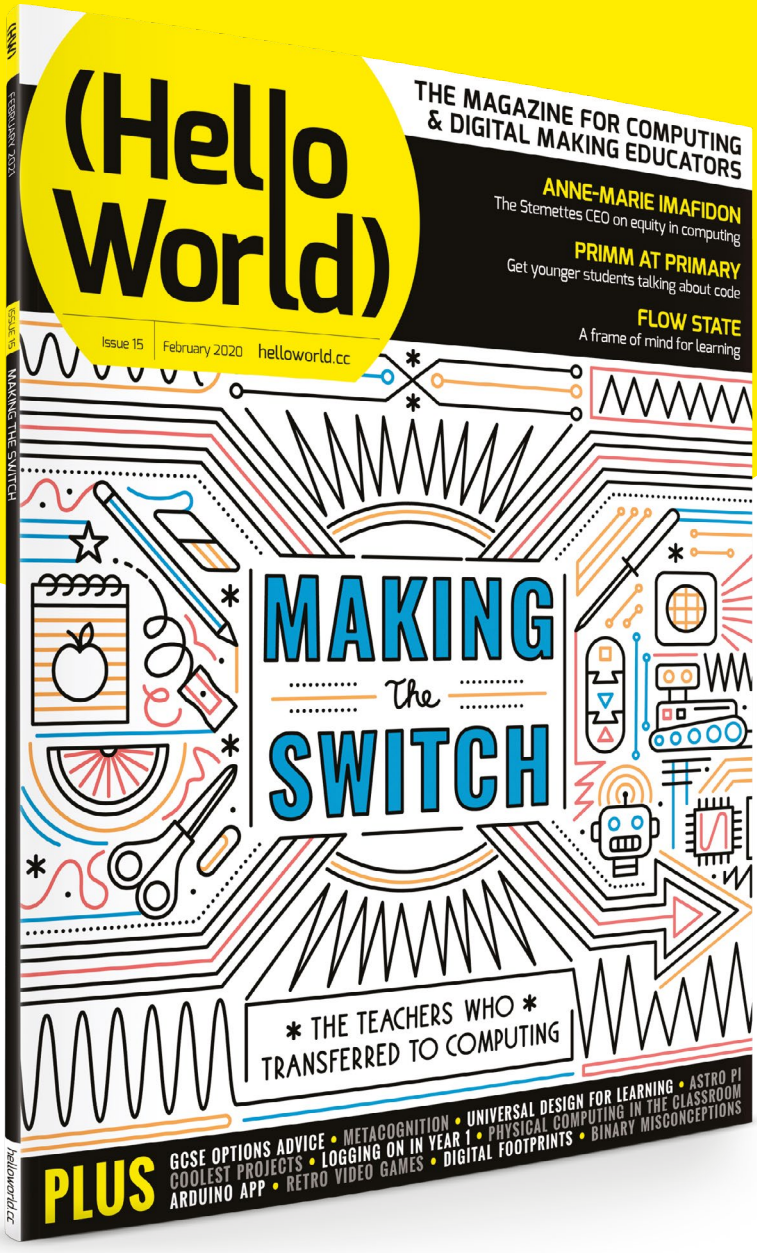
With the subject growing substantially in schools, there is a rising and unmet demand for teachers of computing. I'm part of the team working on the National Centre for Computing Education, which helps teachers switch to computing, irrespective of their starting point. The feedback we have had shows that the training courses and teaching resources on the Teach Computing Curriculum **(helloworld.cc/tcc)** have helped teachers ease the transition. Subject specialisms aren't set in stone — there is always room for growth and evolution. (HW)

DAVE GIBBS

Dave is the educational lead for computing at STEM Learning, part of the National Centre for Computing Education consortium. After working as a science teacher and curriculum leader in ICT, he has supported computing teachers through CPD, resources, and projects since before the new national curriculum (**@adgibbs**).

(HW)

# SUBSCRIBE TODAY




**FREE IN PRINT** for UK-based educators

## Why subscribe?

- Teaching resources and ideas used by over 90 percent of our readers
- Exclusive news, research findings, and in-depth features
- Delivered to your door three times a year


## TO SUBSCRIBE VISIT:

# helloworld.cc/subscribe


## Not a UK-based educator?

- Subscribe to receive the free PDF on the day it is released
- Read features and news at helloworld.cc

**FREE PDF** for anyone, anywhere

FEATURE

# USING THE PRIMM APPROACH AT PRIMARY LEVEL

**Phil Bagge** shares how he implemented the PRIMM approach with his primary school pupils and the impact this had on their knowledge, creativity, and code comprehension

**PRIMM stands for Predict-Run-Investigate-Modify-Make. It is an approach that helps teachers to structure lessons in programming, with each step representing different stages of a lesson or series of lessons. Much as young children learn to read before they write, PRIMM calls for students to read, explore, and understand code before taking ownership of programs and writing their own. It also fosters dialogue in the computing classroom, with students encouraged to discuss programmes and work on them collaboratively.**

The clear steps involved in PRIMM make it particularly useful for teachers who are new to teaching programming. It can also be very helpful for teachers whose students are struggling to understand a particular concept. The approach has already been used by many primary and secondary educators around the world. A study in 2018 with 500 learners aged 11–14 showed improved learning outcomes after 8–12 weeks of programming lessons using PRIMM.

In this article, I will share how I implemented the PRIMM approach with my primary school pupils, and some of the key things I have learnt from the experience. I will go through how I applied each step of the process in my teaching, and how the pupils responded to the different activities involved in each step. I have also created some of my own resources, inspired by the PRIMM approach, which I hope will be useful for primary teachers.

## Predict

In my experience, when teachers are introducing a new programming concept, for example indefinite loops or conditional selection, the first step will be for them to spend time with their pupils ensuring that they understand these concepts away from the computer. This may involve making links with any existing knowledge, or role playing.

The next step will be for the pupils to look at some carefully selected code on paper

| Beachball | Balloon | Bowtie | Bells | Dog |
|---|---|---|---|---|
| Side to side | change colour | ~~Get~~ Get Big + small + spin | Ring | ~~Walk~~ Walk ~~move~~ change to |

A Year 4 pupil's predictions

which uses the new concept, and predict what they think it will do. With my own learners, this prediction has taken the form of a drawing, a summary of what the code does, or an exercise to match the code to a well-prepared teacher summary of what it does. I make this activity low-stakes: it is OK for pupils to get their prediction wrong or partially wrong — especially if they can see why it went wrong in the next stage.

Predicting is a great activity for pupils to do individually, or to work through with a partner of a similar skill level. What I especially like about this step is that it encourages pupils to focus on the wider purpose of the code before going deeper.

> "JUST AS WE READ BEFORE WE CAN WRITE, PRIMM LETS STUDENTS EXPLORE CODE BEFORE THEY BEGIN TO WRITE IT

## Run

Running block-based code examples gives pupils a chance to see whether their predictions were correct. It also gives them time to explore aspects that might not have been included in the paper prediction because of a lack of space. I encourage my pupils to click on individual scripts and run them on their own — this allows them to look for examples of the concept we introduced in the beginning. I find that this step is a great opportunity for pupils to enjoy the program and appreciate its cleverness, which they are probably now beginning to comprehend.

## Investigate

This is the stage at which pupils get to think really deeply about the code. PRIMM builds and draws on other approaches in computing education, including the Use-Modify-Create progression. When first experimenting with Use-Modify-Create, I took the Use step quite literally and asked pupils to use the code without any stipulation, question, or instruction. The most skilled programmers and those who were good at asking questions got something out of the exercise, but the vast majority of pupils made no attempt to read the code carefully or define their own questions. It was not until I considered Use-Modify-Create with PRIMM in mind that I realised that using code needs to involve more than just running it, with the Investigate stage guided by the teacher. I have found that most pupils will not investigate the code carefully without some input and guidance.

In her recent Hello World article, *The I in PRIMM*, the developer of PRIMM, Sue Sentance, explored the Investigate phase of the approach in depth (**helloworld.cc/bm**). In the article, she outlines how the Block Model framework can help pupils to get the most from the Investigate phase by supporting them to thoroughly understand a program.

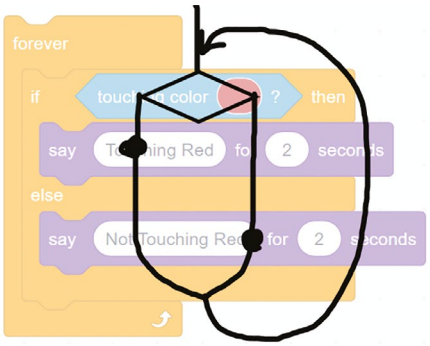
When asking my pupils to investigate their code, I regularly include the following questions:

- How is the newly introduced programming concept being used?
- Can they spot where initialisation has been used, in order to ensure code always operates in the same way, however many times it is run?
- Are they revising concepts covered in previous modules?
- Can they explore the flow of control?

## CODE-IT RESOURCES

The original research for PRIMM is based on text-based programming at Key Stage 3, but I was so impressed by the initial trials of this methodology in Key Stage 2 that I created a detailed free curriculum for primary schools, which draws from the PRIMM approach. This can be found at **helloworld.cc/code-it**.

Many educators from around the world try out my resources and record short sound files saying what worked, what pupils struggled with, and how they have adapted the planning. I really value these. If you use these resources, please get in touch: **helloworld.cc/code-it.contact**.




Drawing the flow of control

I have found that pupils now work much harder in the lessons than I do. This frees me up to work with pairs who are struggling.

I regularly provide reading support if we are using a paper copy of the questions provided in my Investigate and Modify resources, but now that we are using Google Classroom, there are Chrome extensions that will read highlighted text. This allows me even more time for formative assessment.

## Modify

In my experience, many pupils are risk-averse and do not like to attempt things if they are not sure what the outcome will be. Code modification helps pupils to experiment in a safe and structured way. It is a step towards building code creatively, using what is already to hand. Teachers can support pupils to become more creative by asking careful questions which gradually increase in complexity. This aligns with the concept of

## FURTHER READING

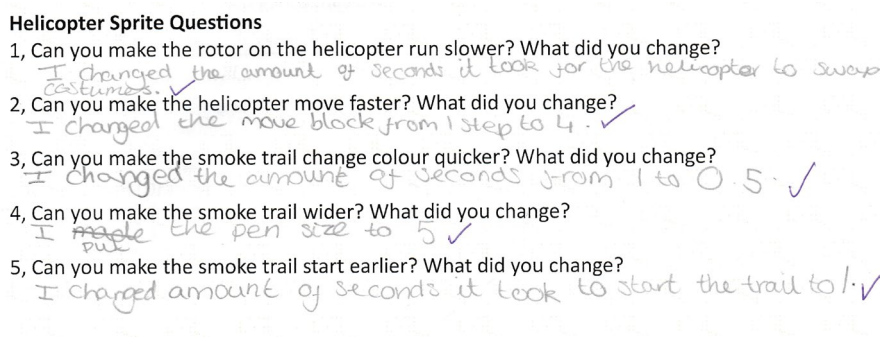

1. Sentance, S., Waite, J., & Kallia, M. (2019) Teaching computer programming with PRIMM: a sociocultural perspective. Computer Science Education 29 (2-3), pp.136-176
2. Teach Computing Quick Read on PRIMM: **helloworld.cc/tc-PRIMM**
3. PRIMM: Something for your programming pedagogy toolkit? Sue Sentance, Hello World issue 4, pp.62-63
4. Reusing familiar techniques. Jane Waite, Hello World issue 6, pp.74-75
5. An overview of PRIMM: **primming.wordpress.com**


the zone of proximal development, whereby a teacher can progress a pupil's knowledge by supporting them through a task that is slightly above their current skill level.

I have found that self-marking the Investigate and Modify phases is an important part of the learning experience for pupils. For both steps, I ask pupils to show me their work if they have got over half of the questions wrong. I always start by asking pupils to tell me about a question they got wrong, to see if they now understand it after they have seen the answer. The vast majority of pupils can describe exactly why their answer is wrong, which leads me to conclude that self-marking is a really important part of the process.

In the Toy Giveaway project, pupils are supported to move from Modify to Make by adding to an existing project template

Asking questions in the Modify phase encourages pupils to understand and learn from their mistakes




> A TEACHER CAN PROGRESS A PUPIL'S KNOWLEDGE BY SUPPORTING THEM IN A TASK SLIGHTLY ABOVE THEIR SKILL LEVEL

### Make

Sue Sentance identified in her Key Stage 3 (ages 11–14) research that moving from Modify to Make can potentially be difficult. I have found the same for Key Stage 2 (ages 7–11) pupils. In my experience, two successful ways to get around this have been to expand the scope and complexity of the Modify challenges, and to give easier initial Make challenges that involve adding things to the existing project studied. In the Toy Giveaway project (pictured), which introduces count-controlled loops, pupils study animated toys that move, change costumes, and so on. For my pupils, a very successful initial Make project is to add another toy to the table and program it using a count-controlled loop. Adding to the template project means that all the code examples that pupils worked with, and thought deeply about, are only a click away.

When I asked them about it, my pupils identified that one of the reasons that they enjoyed using code comprehension methods was that they always felt that they got to make something themselves. Protecting this has been important. If pupils are working slowly and it becomes obvious that they might not get time to make something creative, I typically reduce the number of questions to ensure that everyone gets to the Make stage.

### Final thoughts

Among my pupils, code comprehension methods such as PRIMM have not always been successful in their first few weeks of learning Scratch. I think this is because at that stage, they do not have enough knowledge of the programming environment to investigate or modify code independently. However, once pupils have good knowledge of the programming environment, PRIMM is a fantastic strategy for developing pupil knowledge and agency in the primary classroom, and one that reduces teacher workload for non-specialist teachers. (HW)

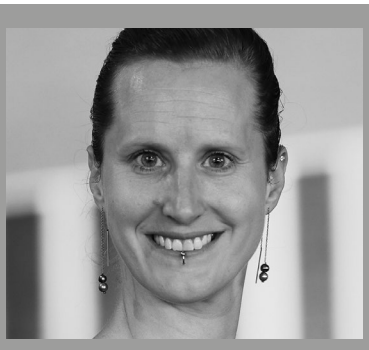

**PHIL BAGGE**

Phil is the computing inspector/advisor for HIAS, the Hampshire Inspection and Advisory Service. He is a teacher at Ringwood Junior School and Otterbourne Primary School. Phil is also a CAS community leader (**@baggiepr**).

# LEARNING TEXT-BASED PROGRAMMING WITH HEDY

Moving from a block- to text-based programming language can be a huge leap for children. That's why **Felienne Hermans** created Hedy, a language that gradually introduces the rules of syntax to young coders

**Block-based languages like Scratch are a great way to help students take their first steps in programming, because they do not have syntax, the precise way in which commands have to be formulated. Students can simply click blocks together and create programs that work straight away.**



FELIENNE HERMANS
Felienne Hermans is an associate professor at Leiden University and also teaches ninth grade computer science. She is the creator of the Hedy programming language, which helps children take their first steps in textual programming (**@felienne**).

## Blocks are for kids

Blocks are certainly great, but when teaching students in seventh grade and up, I find that they start to pull away from Scratch. They begin to see it as a toy; as a thing for elementary school kids. This isn't necessarily true, but around that age students tend to get excited to learn text-based languages, like Python or JavaScript, because they see them as more mature, more powerful, and more real.

## Textual languages are hard

The step from Scratch to Python, though, is quite big. Something relatively simple, such as counting to 10, requires a child to type:

```
for i in range(1, 11):
    print(i)
```

That is a lot of syntax to remember, and a lot of ways to mess up. Misplace a space and Python will crash with a cryptic message called IndentationError. I found that although kids were motivated to learn textual languages, this could cause frustration. And that is not surprising! Recent research on university-level students learning programming showed that even good students make syntax errors in 50 percent of programs. Knowing that, it is not surprising that our middle-schoolers have trouble getting their Python programs to work correctly. I knew I had to come up with a better solution for them.

## English is taught gradually

When I saw the struggles of middle-schoolers learning Python syntax, I started to dive into how children learn to read and write, because when learning a language, children also have to learn syntax rules.

When children learn the syntax rules of natural languages such as English, learning is done gradually. Initially, children just use lower-case letters to make sentences, and that is totally OK! Writing itself is hard enough. New rules are then added step by step: first, children learn that sentences start with upper-case letters, followed by the rule that sentences end in a full stop. Each syntax addition is simple enough, so each step is small, and each new addition gives them new powers. For example, once you know that the full stop separates sentences, you can start to make sentences that span multiple lines!

## Hedy: a gradual programming language

Inspired by the way we teach English in elementary school, I created Hedy, a gradual programming language. Hedy consists of a number of levels, each with a few new commands. In the first level, for example, you can simply print something to the screen with 'print':

```
print hello everyone!
```

Hedy's print statement at level one is a lot simpler than Python, which would require students to enclose the text in both round brackets and quotation marks. Step by step, programming concepts are introduced. Level four, for example, introduces loops, but with a simpler syntax using 'repeat':

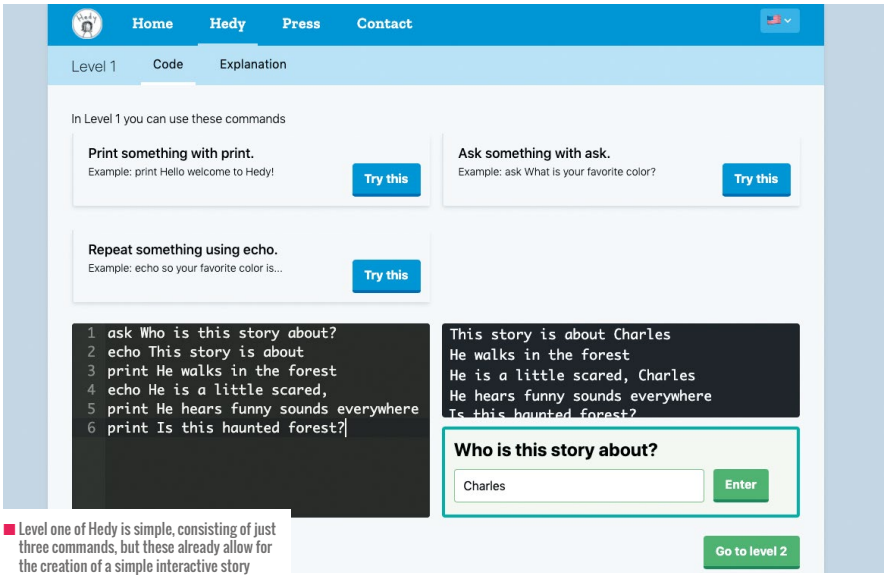
```
repeat 10 times print hello!
```

## Each level is a lesson

Hedy is available for free at **hedycode.com** and can be used in a browser, like Scratch. Currently, Hedy has seven levels available, each lesson having new commands and lesson materials with instructions and assignments built in. Each level should keep children occupied for about one 45-minute lesson. Children can optionally login and save their programs.

## Interaction at each level

Even though Hedy starts simple, each level allows students to make real and engaging projects. For example, level one not only has a 'print' statement, it also supports an 'ask' command, allowing the user to provide input, and an 'echo' command that can repeat the input back to the screen. These three commands enable students to create a simple, interactive story, as shown in the image above. Being able to make something real and fun straight away also helps students to get a sense of where programming could take them. This is especially relevant for children who did not use Scratch before Hedy, and thus might not know why programming is fun!




Level one of Hedy is simple, consisting of just three commands, but these already allow for the creation of a simple interactive story

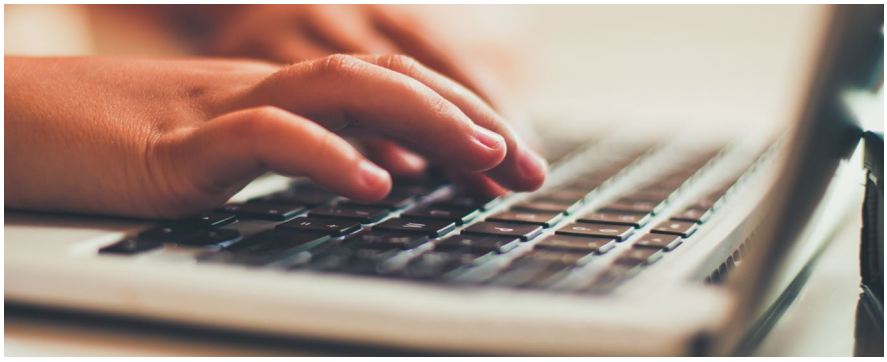
> BEING ABLE TO MAKE SOMETHING REAL AND FUN STRAIGHT AWAY HELPS STUDENTS SEE WHERE PROGRAMMING CAN TAKE THEM

## Explaining why we have syntax

The small steps of Hedy are helpful, but another benefit is that Hedy helps to explain to kids why syntax is even in programming. When I taught Python to middle-schoolers, they would often ask me why, for example, the quotation marks are needed. Usually at that level I would say: "They just have to be there", which I found to be a weak answer, but of course students lack the knowledge to really understand. In Hedy, we first show learners the language without quotation marks, helping them so see the value. In level two, for example, variables are printed automatically, like this:

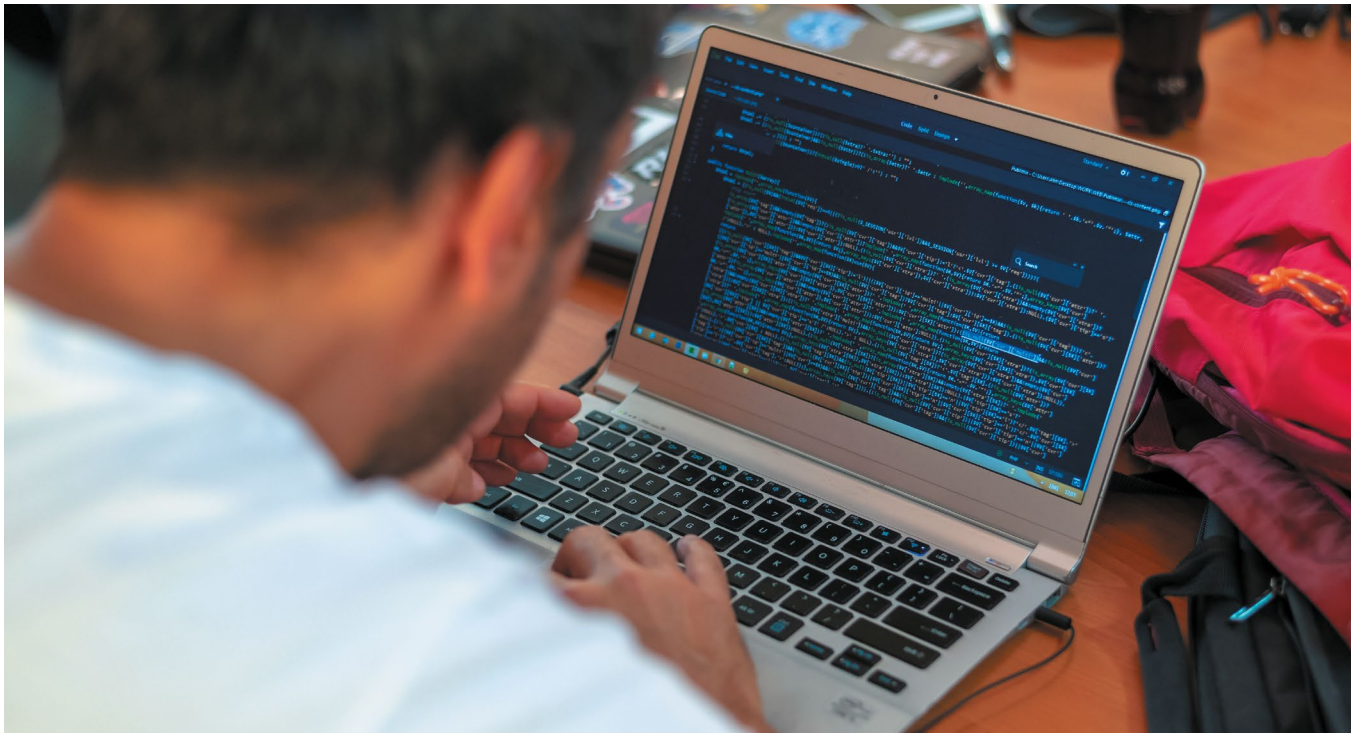
```
name is Hedy
print hello Hedy
```

Being allowed to print variables and text together is easy, but introduces a problem! If you want to print "your name is Hedy" you cannot do it. Typing `print your name is name` will result in "your Hedy is Hedy". This problem is solved in level three, when quotation marks are introduced. This step helps students see why quotation marks are helpful, which is a powerful realisation.



## Better error messages

One of the big struggles when using a textual language is the error messages. When you make an error in a print statement in Python, you might be confronted with cryptic messages like 'SyntaxError: unexpected EOL'. Such messages can be discouraging for new programmers, because they are hard to



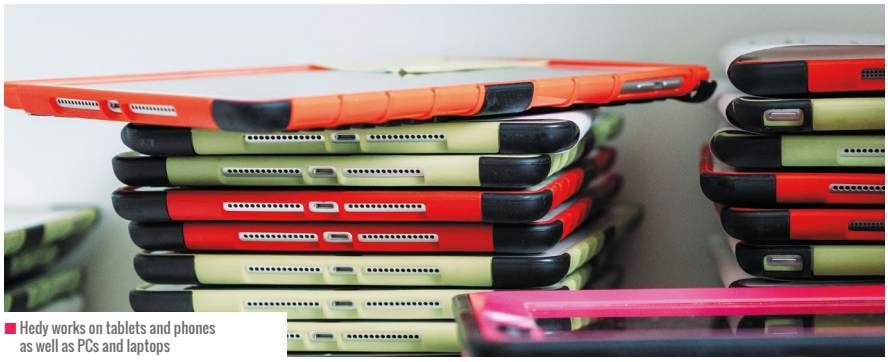
understand, and also not actionable — how should we go about removing this unexpected EOL? As Hedy has a simple syntax, it is also easier to guess what the child meant. For example, if you misspell 'print' in Hedy, the error message will read:

```
pinrt is not a command in Hedy
level 1, did you mean print?
```

## The future of Hedy

Hedy is free to use, and over the last nine months more than 200,000 programs have been created. Hedy can be used for a diverse range of programs, and we have seen children who really go wild with creating 100-line programs in a level before advancing to the next. We see this as a strength, because it can take a while before you really get the hang of syntax. So a lot of practice at a lower level will contribute to your programming skill later on.

There is, of course, more work to be done. We are now available in English, Dutch, French, Portuguese, and Spanish, but we'd love to add more natural languages so that we can reach more children. We also want to explore adaptive levelling. In the current version, children can choose to move on a level, which they sometimes do after just one program, because they are curious or ambitious. It would be better to move them on automatically after they have shown enough skill, or maybe even move them back when we see them struggle. We are excited to see what young people create next with Hedy at **hedycode.com**! (HW)



Hedy works on tablets and phones as well as PCs and laptops

## A TEACHER'S EXPERIENCE WITH HEDY

Ramon Moorlag, a computer science teacher in the Netherlands, explains how his students have used Hedy:

"Hedy is fun to use. Taking small steps, focusing on a single concept and not being distracted by conventions and/or typos helps a lot.

"My seventh graders enjoyed Hedy. After explaining the lesson concept and practising it, I ask students how to use the concept in a free form. They have built all sorts of things because it helps students create interaction from the start. For example, I have seen students build knock knock jokes, jinxes, riddles, and interactive Christmas wish lists.

"So it's not only a way of bridging the gap; it's also a great way to engage students in textual programming.

"After running the Hedy course for the first time, I found that they had a lot more confidence when we started with some simple HTML and web tutorials."

# HOW MODELLING CAN SUPPORT LEARNERS

**Josh Crossman** explains the modelling approach: by demonstrating a new concept, teachers can support their learners and develop their own understanding of the key skills and materials being introduced

**When giving learners the opportunity to use new skills or software, it is important to show them how first. Teachers can demonstrate a new concept through a recorded video, or by modelling it for the learners. Modelling is an instructional teaching strategy, through which a teacher demonstrates a new skill or approach to learning. Teachers first model the task or skill for learners, and then learners begin the task and work through it at their own pace.**

Schools use modelling across many disciplines, such as writing, handwriting, mathematical strategies, and science experiments. It's a powerful strategy that can be used across many different subjects, and computing is no different.

As we have been developing units for the Teach Computing Curriculum (**ncce.io/tcc**), we have been reflecting on the tried and tested techniques that help make computing lessons a success. In this article, I will share some of our top tips for modelling.

## Improved confidence

Perhaps the biggest benefit of modelling is the confidence and competence that teachers gain from using the software. In the Teach Computing Curriculum Year 6 (ages 10 to 11) 3D Modelling unit, learners are shown how to create a 3D shape and change the viewing angle within the 3D modelling software Tinkercad. For teachers who are new to 3D modelling, changing the viewing angle may be something that they have not come across before.

Through pre-lesson preparation, and of course, through using the software when modelling, teachers may encounter misconceptions, errors, and perhaps shortcuts that the learners might make in their own use. For example, in Tinkercad, teachers can click on the viewing cube to jump to different views. As their own confidence improves, supporting the learners with their errors or misconceptions will become easier.

## Thinking aloud

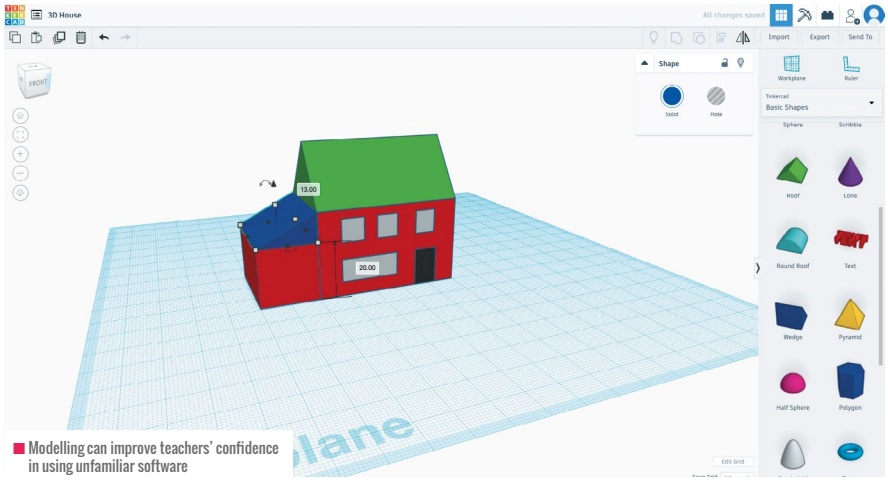
If teachers provide a monologue — or 'think aloud' — as they model, learners get the opportunity to observe expert thinking that they wouldn't usually have access to. It allows learners to follow more closely what the teacher is doing and why they are doing it. More precisely, it can reduce the extra cognitive load of having to deconstruct each step for themselves. More information about cognitive load theory is available at **helloworld.cc/cl-theory**.

As teachers know their own learners best, they can tailor the language they use when modelling, to make it as beneficial as possible for their particular learners. This ensures that teachers can meet their learners' needs in a more targeted way.

## Greater freedom

Questioning is a key part of modelling — this includes both the teacher's questioning of their learners to draw out understanding, and the learners' questioning of the skills and processes the teacher is using. Modelling can enable teachers to give context to their answers to the questions, ensuring they are able to 'show' alongside the 'tell'. This will help to make their answers more visual and concrete.

Additionally, the freedom of modelling



Modelling can improve teachers' confidence in using unfamiliar software

ensures that teachers are able to meet the needs of all the learners in the room. This could mean revising an area that wasn't fully understood by some, or perhaps moving at a slower or quicker pace, depending on the learners' level of understanding. For example, in lesson one of the 3D Modelling unit, teachers could break up the content in the video into smaller chunks, perhaps allowing the learners to get to grips with the viewing angle before introducing the zoom buttons.

## Impromptu learning

In line with the benefit of greater freedom discussed previously, modelling also allows for more impromptu learning. As students ask questions, teachers may need to traverse different areas of the software, or challenge new thinking around a skill they hadn't previously thought about. Using the freedom provided by modelling can lead to a deeper level of understanding. Equally, it can reinforce the skills used in previous lessons, giving learners the chance to consolidate that learning as they progress to the next stage.

As teachers narrate their modelling, it can also bring to the forefront things they do instinctively. Learners can pick up on these things, such as keyboard shortcuts (copy, paste, and duplicate come to mind!) that teachers might use naturally. Modelling can also lead to exploration of a skill that might benefit learners but wasn't originally planned for the lesson. These are things we do all the time when using new software or learning new things, but we might not always use the opportunity to share them.

## Making mistakes

By demonstrating computing experiences live with their learners, teachers can encourage the sharing of mistakes, both intentional and unintentional. Making and demonstrating mistakes is OK — in fact, it should be celebrated! Allowing learners to see that their teachers aren't immune to mistakes can be encouraging for less confident learners and help them to build resilience. More importantly, it enables learners to see how to respond to mistakes, particularly when combined with the 'thinking aloud' approach mentioned previously. This can be a powerful tool for encouraging perseverance.

By sharing mistakes and thinking aloud, teachers can guide their learners through strategies to overcome a range of obstacles. Use phrases such as: "When I first looked at this problem, I didn't know where to start" and, "It's OK to feel frustrated at this point; I often do."

## Video demonstrations vs live modelling

As well as live modelling, there is also the option to demonstrate software using prerecorded videos. This means that you can show the video to learners and they can follow along and learn the skills you need them to know. However, many of the benefits to modelling, such as impromptu learning and improved confidence, will not be gained in this way.

Recorded video definitely has a place in the classroom — particularly in the current teaching context — but modelling these skills yourself, making mistakes, and developing ideas as you go are invaluable ways to share learning experiences. If you do show your learners videos instead of using live modelling, here are some tips for including the modelling approaches:

- Watch the video before the lesson and practise the thinking aloud approach
- Share the video with the learners and think aloud as the video is shown; allow the learners to attempt the task before asking them to share their own modelling of it
- Punctuate the video with your own questions, and don't be afraid to pause to answer questions posed by the learners

Throughout our work on the Teach Computing Curriculum, my colleagues and I recorded videos to model key skills. Our aim was to ensure that the content was as accessible as possible for the full range of teachers who may use it.

Though modelling may take longer than displaying a prerecorded video, it is a great way to share the learning experience. In my view, it has many benefits over the more passive prerecorded approach. Hopefully, many teachers will watch these videos and gain the confidence to give modelling a try.

Modelling is a journey, and teachers inevitably won't get it right every time — but sharing that experience with the learners may turn out to be the most powerful part of the exercise. (HW)


**JOSH CROSSMAN**

Josh is a programme coordinator at the Raspberry Pi Foundation, working across programmes such as the Teach Computing Curriculum and Hello World. He is a Raspberry Pi Certified Educator and a former primary teacher.

# USING TABLETS TO ENHANCE CROSS-CURRICULAR LEARNING

**Charlotte Spenceley** shares her experience of incorporating tablet computers into her teaching approach, and how this has benefited both her and her pupils

**I am fortunate to work in a school where significant investments have been made to promote and develop the computing curriculum — and one key benefit is that every pupil in Key Stage 2 (ages 7–11) has their own tablet.**

Nearly two years ago, I was given the opportunity to work with an experienced computing specialist to develop a cross-curricular approach to using tablets in the classroom. This opportunity came about through fortunate timetabling and as a result of my reputation for being interested in using technology with my students — I'm one of a small number of people who know how to get the interactive display and sound in the school hall to work at the same time! The specialist, who was known to our head teacher, was employed at the school to lead the computing curriculum, provide our pupils with stimulating opportunities, and deliver continuing professional development for staff.

When implementing the use of tablets in the classroom, our aims were for pupils to develop key literacy skills, explore new routes for creativity, and learn how to record and evaluate their work.

Because of these investments in developing the computing curriculum, I have learnt more about using technology in the classroom in the past 18 months than I have in almost ten years of teaching experience. Prior to this, my experience of computer science was very limited and I had scarcely been exposed to computing or technology in the classroom. Tablets were my starting point, and I had one afternoon a week with our specialist and a science curriculum to adapt around the topic of rocks. We began our journey with this topic for timetabling reasons, but quickly adapted the lessons learnt to other areas of the curriculum, such as history and English.

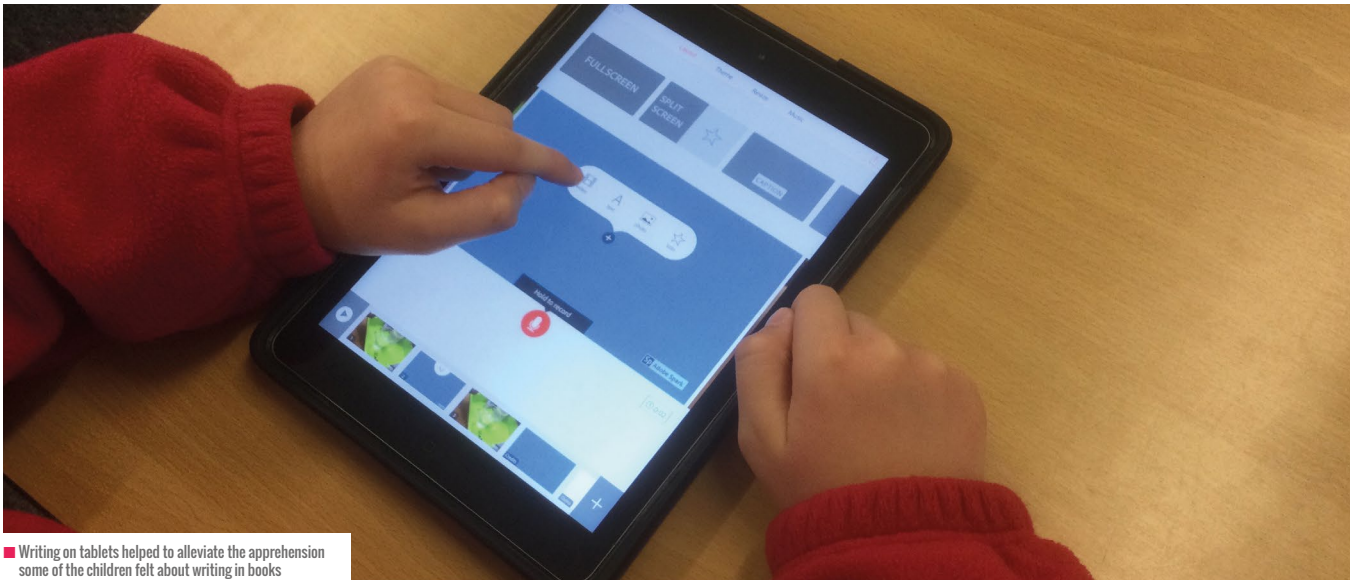
## Getting started with the tablets

Our journey began with eliciting the pupils' understanding and curiosity by using the PicCollage app to write questions over a photograph of a rock. Straight away, my students and I encountered problems: we couldn't get the camera covers off the cases; we didn't know how to capitalise letters; and we were afraid to press any

### INTRODUCING TABLETS TO YOUR PUPILS

- Let children explore the apps
- Teach them what key symbols mean, for example, press ✚ to add something
- Introduce one skill at a time


Using Adobe Spark Video, we were able to bring to life an investigation of the permeability of rocks




Writing on tablets helped to alleviate the apprehension some of the children felt about writing in books

> ONE OF THE MOST POSITIVE CHANGES IN MY PUPILS IS THE ABILITY TO REFLECT ON THEIR OWN EVOLVING KNOWLEDGE

buttons for fear of "destroying the tablets" (to use the children's words). I questioned myself throughout: was the time used to model every tap worth it? Was every child making progress from their individual starting point? What was the difference between the tried and tested strategies?

Despite these initial hurdles, the dialogue created through this activity allowed for thoughtful, memorable, and reflective learning among the group. Writing on the tablets also helped to alleviate the apprehension that some of the children felt about writing in books. Misconceptions were identified, understanding was assessed, and I was able to reflect on my teaching. As with most things in a school environment, I learnt that patience needed to be embraced, and that the quality of participation, confidence, and problem-solving skills witnessed in one afternoon justified the time needed to embed routines. Most importantly, I learnt that children need to be taught how to use technology in a meaningful way, and not to assume that all young people are digital natives — a term often used to describe the generation that has grown up in the era of technology and the internet.

## Growing more confident

Eager to build on our initial exercise, our subsequent activities explored different methods of recording. A popular app with my class is Balloon Stickies Plus, an application that allows you to add speech bubbles to photos. Not only do they like the name; they also enjoy being able to choose between typing and voice recording, and the freedom to present their work the way they want to. I have found that taking away the structure of an exercise book has allowed children to focus on the content and creativity of their learning.

One of the most positive changes I've seen in my pupils is their ability to reflect on their own evolving knowledge. Using Adobe Spark Video, we were able to bring to life an investigation on the permeability of rocks. We used technology to look back at prior learning, and this helped to predict and justify the results. As each experiment was conducted, the tablets were used to record the results visually. Spark Video supported our observations, reasoning, and conclusions by allowing us to explain each step verbally and annotate the video with additional information. Having to record experiments by written methods can sometimes hinder engagement and enthusiasm. However, by patiently developing the skills needed to use tablets as an effective learning tool, children can grow more confident with taking risks, and become more engaged, reflective, and digitally literate learners.

## Overcoming barriers

There are still some barriers I am working to overcome: balancing book work with technology; figuring out how to show constructive marking and give feedback via the tablets; and planning the next steps in embedding the cross-curricular use of technology within my setting. However, my pupils are becoming confident, reflective, and enthusiastic learners (to use their own words again) and I believe that this shows the strong impact that tablets can have when used as a learning tool in the primary classroom. (HW)

CHARLOTTE SPENCELEY

Charlotte is a Key Stage 2 teacher and school council lead at Giffard Park Primary School. She is also a newly qualified teacher (NQT) mentor and is the health and well-being champion at her school.

**DAHLIA POULTON-TRASK** STUDENT

# GIVE US CONTEXT AND CREATIVITY IN COMPUTING

With gender balance in computing an urgent issue, **Dahlia Poulton-Trask** suggests that students need to be shown the scope of possibilities with the subject

**I am an 11-year-old student and at the moment, everyone in my year group at school spends the same amount of time learning about computing. However, statistics show that when it is time for us to choose what we would like to study at GCSE and then A level, most girls will choose other subjects over computer science.**

While things are improving — in the UK, the number of girls taking computer science at A level has increased by 300 percent since 2015 — there's still a long way to go. And the problem definitely isn't that girls aren't as good at computing, because we know that in 2019, 34 percent achieved A grades or higher at A level, compared to 26 percent of boys.

I love computing, because at heart it is all about solving problems — either small problems like debugging a 'for loop', or huge problems such as predicting Covid infection rates. But I feel that there's a big opportunity to make computing lessons at school more creative and more meaningful. I know this would make me enjoy the subject even more, and I think the same would be true for lots of other girls, too.

I would love to analyse the code that makes a playlist in Amazon Music or Spotify, or adapt code that can be used to create cybersecurity algorithms, such as code that checks passwords. In class we are often learning how to code with no context, and we don't hear much about coding in our other lessons, which means we often don't learn about the really inspiring things that you can do when you learn computing.

I recently spoke to Rina Lai, a researcher in educational psychology at Cambridge University. She agrees that "gender imbalance in computing has been a very large and urgent issue" and that getting more girls into computing "relates to a lot of things, including how computing is being introduced to them at school, and their motivation, as well as their personal interests".

Another interesting perspective from Rina is that it's important to promote computational thinking. Solving problems, across many different subjects, is computational thinking, and I think understanding this would help more girls to realise how amazing computing is. If schools are teaching grid references in geography, for example, they can use Google Earth or Maps to support this work, but they could also look at how following complex maps is a computational thinking skill, and I think everyone would then make the connection to computing more easily.

Teachers could also help get more girls interested in computing by setting tasks in which students help each other to learn. If girls start making tutorials on how to code, they might get other girls interested. This is partly why I decided to set up the YouTube channel Coding Siblings (**helloworld.cc/codingsiblings**) with my brother Ethan. We launched the channel in November last year and we have created a lot of short videos about the main concepts of programming in Python that are short and easy to follow. I hope that sharing these videos helps get more young people, and girls especially, interested in computing. (HW)

DAHLIA POULTON-TRASK (**@CodingSiblings**) is a Year 7 (ages 11-12) student at Cardinal Newman Catholic School in Hove. She likes reading, gymnastics, and solving problems in code.

# CREATIVE WAYS TO CONNECT

A CoderDojo club in India overcame significant challenges to deliver unplugged coding activities and Python training to young people

**Our CoderDojo journey began unofficially in February 2020 when some friends and I organised a five-week Scratch workshop. We offered the workshop to students who we were teaching at weekends under our Joy of Helping programme — an educational initiative to build creativity, confidence, and linguistic and scientific skills in children. And then, of course, Covid-19 happened, and many of us went into online teaching mode.**

Since then, our CoderDojo journey has taken place in the world of the pandemic — and with that has come many challenges. It was definitely not easy to bring all of the students into an online class environment. For many families in our community, having a touchscreen phone with an internet connection is a luxury, so getting access to a device was an immediate challenge for lots of our young students; even having a phone in the family was not a guarantee that a child could join a session, as their parents might leave for work in the morning and take their phone with them.

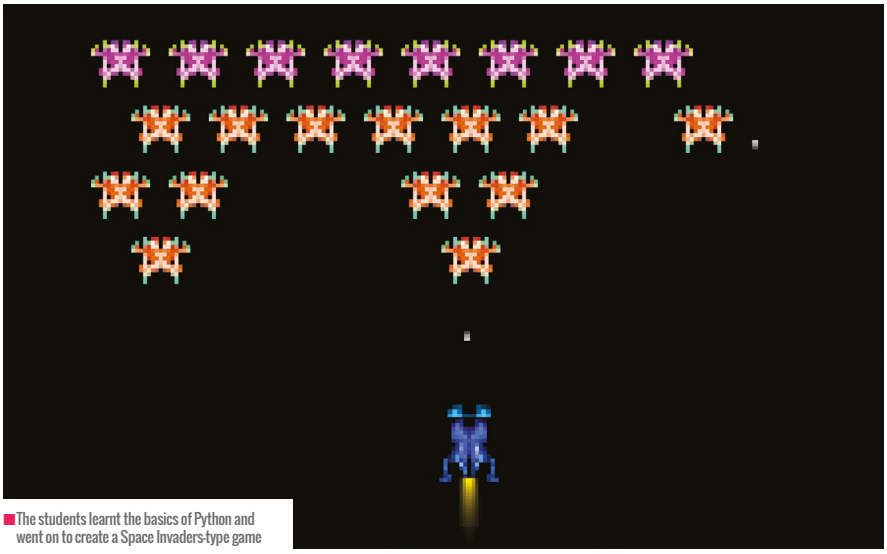
With these barriers in mind, we delivered our online sessions through two separate programmes at weekends: unplugged activities and live Python sessions.

## Unplugged activities

Many parents and volunteers in our community use WhatsApp, so we knew that this would be an effective way to distribute resources to our young people. We began by circulating worksheets for unplugged activities within our WhatsApp groups, giving students a week to attempt the activities and ask questions. Slowly, more and more students began to really enjoy these worksheet activities, and in July we began online sessions on Sundays to meet this demand.

In the unplugged activities sessions, we usually start by having a discussion with the kids on a topic such as debugging. We spend time understanding what debugging is, why it is important, and how we can do it. Then we work on an exercise together and help the kids to solve it. At the end of each class, we give the students another exercise as homework, to complete for the following week. We have found that this approach works well for students with limited access to a device, as they can learn computing concepts in their own time by doing the unplugged worksheet exercises.

We have also done unplugged activities such as logic number sequences, block logic puzzles, and sudoku challenges. One unplugged activity that worked particularly well for us was relay programming, where we used arrows to race through a paper graph. Code.org has some useful resources for teaching relay programming (**helloworld.cc/relay**). The aim of these activities was to develop skills like debugging and understanding of sequences, and also to help children learn computational thinking skills. We were amazed by the curiosity of the young people, and how after four or five sessions the speed at which they could solve problems increased. They also became interested in doing more such activities. In the beginning, we had to give some pointers on how to solve certain problems, but as time went by, they needed our help less.

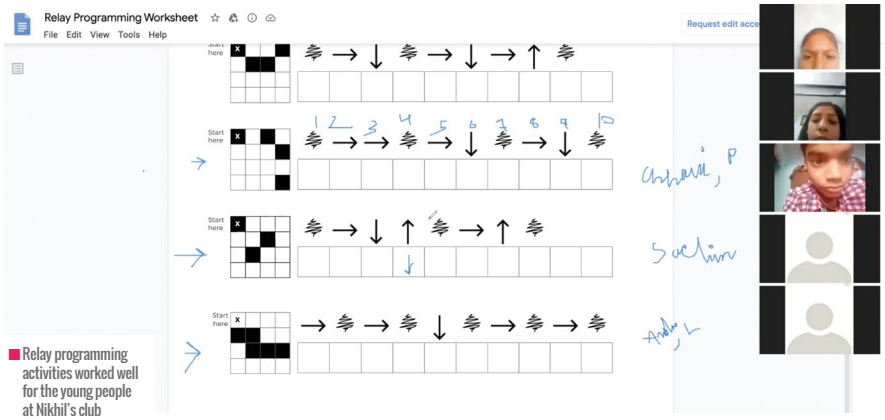


The students learnt the basics of Python and went on to create a Space Invaders-type game

Since then, we haven't looked back, and we have also introduced Microsoft Word and Scratch coding in these sessions. We still face logistical challenges — such as limited access to computers for our students — but we try to keep learning with an optimistic and positive attitude.

## Live Python sessions

We started our second programme in September 2020. The aim of this session was to teach Python to the enthusiastic young members of our community.



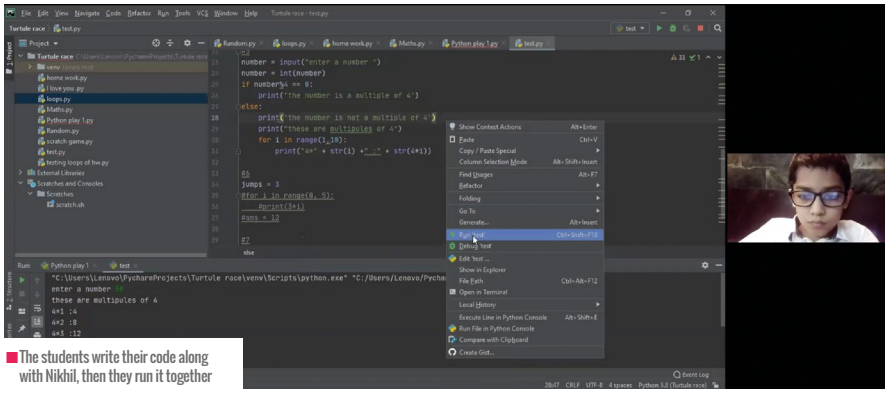

Relay programming activities worked well for the young people at Nikhil's club

> WE STILL FACE LOGISTICAL CHALLENGES, BUT WE TRY TO KEEP LEARNING WITH AN OPTIMISTIC AND POSITIVE ATTITUDE

In our first four Python sessions, we began by covering the basics, such as variables, strings, loops, and conditionals. We also spent a good amount of time on learning about the indentation, syntax, and set-up of a new project. Once we had finished learning the basics, we were able to move on to creating games — and we have just finished our first Python Space Invaders–type game. Each Python session is two hours long with a 15-minute break. We encourage the young people to stretch in between the breaks, as otherwise they are sitting still for a long time.

Each student joins the session via Zoom. I usually begin each session by first discussing the logic with them. Once we all agree on the logic, I go on to implement it. We all get to work at the same time; the students write their code along with me. We then run our code together, discuss any issues we've found, and make any required minor changes. Sometimes we share the code with the group using the Zoom chat function. From time to time, I ask each student to share their screen, show the output of their code, and explain the logic they have used to the rest of the group. This is a good way for me to check on how a student is doing, and also gives them an opportunity to share their progress and develop their presentation skills.

The experience of teaching such curious and passionate students has motivated us a lot. We faced some logistical challenges with our students, such as the installation of Python and IDEs, but the Zoom remote access and annotation features came to our rescue. We have learnt that being extra patient, along with taking short breaks in between classes, can help us to teach quite effectively.



The students write their code along with Nikhil, then they run it together

## Plans for the future

We want to keep the momentum of our CoderDojo club going; in 2020, we were able to register as an official club with the help of Vasu, the club programme coordinator for India. We plan to continue our unplugged sessions for at least four months, as our students like these sessions a lot. These activities will also help us to build a strong foundation of computational thinking in our students. After that, we will focus on learning Microsoft Word, Scratch, and HTML. We are also planning to organise a collection drive for old laptops and mobile phones for our students.

In our Python sessions we plan to program a few more games. We want our kids to learn that some hard work and effort is required to get comfortable with a new programming language. We are also planning to create some physical computing projects using Raspberry Pi. (HW)

NIKHIL HANDA

Nikhil is a software development engineer at Adobe Inc. He became involved in the CoderDojo movement in early 2020 and has a keen interest in teaching students programming and computational thinking skills. Nikhil runs a CoderDojo club in Faridabad, in the Indian state of Haryana (@NikhilHanda43).


These Heart of Midlothian FC players are wearing technology vests during training to track their performance and fitness levels

# BUILDING DIGITAL SKILLS THROUGH SPORT

**Tanya Howden** shares how one club in Scotland is harnessing the power of sport to teach young people about technology, challenging stereotypes along the way

**Heart of Midlothian Football Club is a professional football club in Edinburgh, Scotland. In 2019, we at the club launched our Innovation Centre, an initiative that aims to support the local community in developing important digital skills for the future. We work in partnership with schools and leading industry experts to design and provide a unique array of free digital education programmes for young people. They support young people to discover how to use technology creatively to solve the problems that are important to them.**

We use the power of sport to engage a wide range of young people to come along to the club. For some, this is the first time they have taken part in a coding club. While many of the young people who attend the club dream of one day becoming football players, we show them some of the many other ways in which you can work in sport — with technology being a very exciting area that is only going to continue to grow.

We launched the Innovation Centre with support from Baillie Gifford and Dell Technologies. As support from local delivery partners, schools, and industry experts has grown, we have continued to introduce new clubs and programmes to the community that explore different areas of technology in the same unique learning environment.

## Build your own fitness tracker

It's great to incorporate movement and exercise into lessons, as it's an opportunity for young people to burn off some energy, and helps them to focus on the project they are working on. Our club attendees' favourite project is always the fitness tracker, created using sensors in the micro:bit. It's important to test our creations, so we always bring everyone in the group together to test their newly created fitness trackers by seeing who can do the most star jumps in 30 seconds.

For more information on making your own fitness tracker, the micro:bit team has put together a handy guide, available

## BEHIND THE SCENES

When thinking about sport, you might first remember a recent game you watched or the latest results for a team you support. When considering the technology used at sports stadiums, your thoughts might then turn to the ticketing system used or the ways in which clubs can engage with fans online each day.

Aside from the operational side of a sports club, we see that technology is used on the pitch to support sporting officials with decisions they make during a live game. Goal-line technology uses several high-speed cameras located around the pitch to detect the ball's exact position as coordinates, to determine whether there has been a successful goal. Referees may also look to VAR (video assistant referees) to help confirm a decision by reviewing video footage of the game, usually in slow motion to break down the player's movements.

Other sports use similar systems, such as Hawk-Eye in tennis, used to determine whether the ball was within the lines of the court. There is even a rugby club in England that uses virtual reality during training with players to explore and learn about handling different scenarios.

Sports teams will also make the most of the technology available to them to track player performance and fitness. In football, most clubs will use technology vests with sensors that track and capture data from players during training and match days. Examples of the data captured includes how fast players are running, or their location on the pitch at different times.

The coaching team can use this information to help make decisions such as the team line-up, player positioning during the game, or fitness routines that players should be following. The amount of data captured in sport is astounding, and the insight that it can give teams is vital to planning for future games and tracking individual and team progress.


The club sessions took place at the media suite in the stadium, before moving online because of the coronavirus pandemic

> WE LOOK AT DIFFERENT WAYS IN WHICH TECH IS USED IN SPORTS AND RECREATE OUR OWN VERSIONS USING THE MICRO:BIT

at **helloworld.cc/fitness-tracker**. To track movements other than steps, you can also explore what other gestures and movements the micro:bit can recognise.

### The power of sport to engage

We run our digital education clubs in the media suite at the stadium. This is the same room that on match days is full of journalists reporting on the game happening, or getting the latest interview with the team manager. Regardless of whether attendees are interested in football, being in this space is very exciting and creates a real buzz in the room.

Our CoderDojo club — which we usually run in six-week blocks — was the first club to be introduced at the Innovation Centre. Each week we look at a different way that technology is used in sports, using the micro:bit to recreate our own versions of devices such as fitness trackers, as described in the previous section. As with our fitness trackers, everyone starts by creating the minimum required to have a working project, with young people then customising their projects and adding on features based on their own interests.

At the end of the programme, we invite parents to come in and hear from the young people about the projects they have created in a show-and-tell session. The confidence of every young person as they talk about their programs creates a contagious excitement in the room.

We see almost all the young people returning to the stadium to take part in the same club again, or attending one of the other clubs we run, such as our robotics club, at which young people code their own robot football players. We have even had some attendees return as youth mentors to support sessions for primary school learners.

> WE PROVIDE AN ARRAY OF FREE DIGITAL EDUCATION PROGRAMMES TO GET YOUNG PEOPLE ENGAGED WITH TECHNOLOGY

## Challenging stereotypes

The Innovation Centre aims to challenge stereotypes that girls and boys may have about careers in coding and engineering, using the hook of football to engage young people who might not otherwise have become involved. While some young people attending the club will already be engaged in these subjects at home or school, for the majority, this is a new area for them to discover. To begin with, confidence is often low, with young people being concerned that they won't be very good at coding. This is never the case.

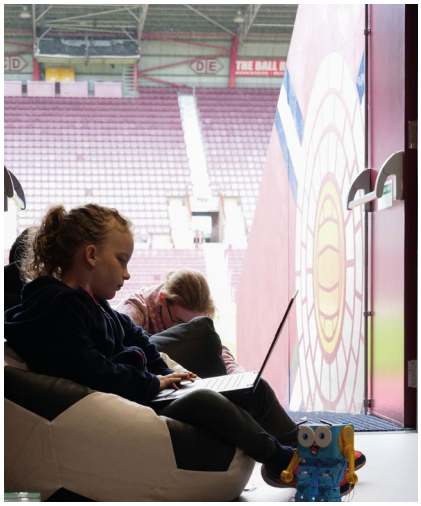
Like coding and engineering, sports can be seen as something that is just for boys. We want to challenge both of those stereotypes in our digital education clubs, and regularly have visits and talks from players in the women's team. We have a diverse set of volunteer mentors in each session and are close to having a 50/50 gender split in our clubs. Teachers may find it useful to research different women engaged in sport, to show their students some new role models.

## Finding new ways to learn

Just like every organisation, we have had to adapt our activities to align with local coronavirus restrictions. While we were unable to run any clubs in person, we had families who regularly attended our digital education clubs and were looking for activities that their young people could do from home to keep learning.

Using Google Classroom, we set up online clubs for learners in primary and secondary schools (called Digital Athletes and Digital Champions respectively) that hosted lots of different STEM-related activities, with a focus on staying active and exercising. Examples of these activities include designing digital posters about the importance of staying active at home, or coding beat the goalie games online.

Girls and boys alike are encouraged to explore technology through the lens of football



Making our clubs available online allowed a wider group of young people to get involved; we had over 200 families register to take part from across the UK. They would also regularly send us photos and videos of the activities that they had completed. We were overwhelmed with the response, and although there was a steep learning curve to overcome, we are really excited to continue exploring how we might be able to engage with more families and organisations across the country in the future. (HW)


### TANYA HOWDEN

Tanya is the digital education programme manager at Heart of Midlothian FC. With a background in computer science, she has been running CoderDojo clubs for over four years and is interested in showing young people – especially girls – all the different and creative ways in which technology is used (**@tanyahowden**).

# CODING AT HOME WITH YOUNG PEOPLE ACROSS THE WORLD

Digital Making at Home's **Kevin Johnson** shares ideas for designing and delivering successful online teaching content

**In March of last year, day-to-day life as we knew it changed dramatically. At the Raspberry Pi Foundation, we realised that we needed to act fast so that young people still had access to digital making learning experiences while learning remotely. The result was Digital Making at Home, a collaborative and evolving repository of content and resources which young people — as well as teachers and parents — could access for free to enrich their computing education.**

We knew that it needed to be a programme that was easy to dip into, that was about having fun first and then learning along the way. There were bumps in the road in developing and delivering Digital Making at Home, but we learnt a lot and the programme continues to evolve as a result. Much of what we learnt can also be applied to a virtual school environment. We would like to share what worked for us to help educators deliver engaging digital making experiences online.

## From classrooms to living rooms

We began by immediately putting together a team who started writing content that could be taught via video. This content drew from our existing library of resources in a new way; feedback we received from parents, friends, and families in our community indicated that they found the content more relevant and accessible in its new format. Ryka, for example, is a ten-year-old who attends a CoderDojo club. Her parents told us, after discovering Digital Making at Home: "We've noticed that she is engaged and takes interest in showing us what she was able to build. It has been a great use of her time."


Young people can enrich their computing education with Digital Making at Home

Digital Making at Home was launched at the end of March as a weekly blog post that included three code-along videos for different skill levels, led by members of our team — who were sometimes coding with their children as well.

We focused on a different theme each week. Having a weekly theme allowed us to explore different real-world topics, many of which can be directly linked to school curricula, such as well-being, space, and the environment. Our hope was that if we made the learning experience more meaningful to young people, they would be better motivated to create.

Each week, we encouraged young people to share what they created with us through a contact form, and we featured a selection of them in a weekly project showcase blog post. The young people loved this, and we saw consistent engagement from a handful of coders who would excitedly share projects with us every week.

We would certainly recommend giving young people an opportunity to showcase their work while they are learning from home. In the absence of an in-person session, we found that it really helped boost young people's confidence and made them feel that they were part of a fun, global community. Many young people who showcased their work for us were coding at home with a parent or a sibling, and it was great to see this wider network included. This could certainly be replicated in a virtual classroom environment for students who would like to share something cool that they've made.

## Introducing a weekly live stream

A few weeks into the programme, we decided to launch a live video stream so that learners could engage with us synchronously. The world was missing routine, so providing a weekly activity that viewers could look forward to felt like the right thing to try. The stream also acted as a temporary substitute for in-person engagement, which gave us the opportunity to invite special guests from across the tech community to join the stream. We welcomed a huge variety of special guests, including animators, engineers, CoderDojo volunteers, past Coolest Projects participants, and colleagues from the Raspberry Pi Foundation. Inviting special guests to present a talk or hold a workshop may also work well for teachers who are hosting live classes with their students — it is also a great way to show young people how computing is relevant to many types of career.

Over the next 32 weeks, we produced 70 videos and over 45 hours of content, with 40,000 viewers. Even with this data, there was still a lot we didn't know, such as who was watching, and whether viewership was linked to the decisions we were making. We began placing feedback surveys in the live stream YouTube descriptions and on our blogs, and those provided great qualitative feedback. One of our adult learners said the videos were "very easy to follow" and that "the explanations are really well done for each step". Much of the feedback we received made it evident that our audience had taken a liking to our hosts, Mark Calleja (Mr C) and Christina Foust.

We also reached out to members of our club programmes to find out what they were doing and how they were approaching at-home learning. We got a lot of positive feedback from those conversations, and some really helpful suggestions as well. For example, a Raspberry Pi Certified Educator, Yolanda, suggested that the live streams be no longer than 30 minutes, and highlighted the importance of her kids being able to pause and rewind while coding along with us.

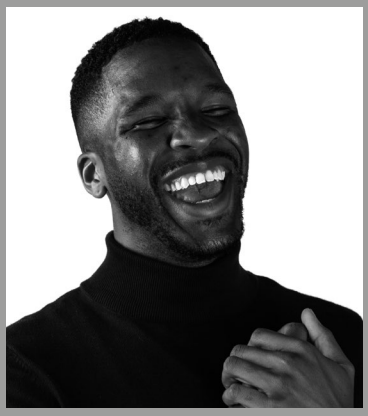
> "A HANDFUL OF CODERS WOULD EXCITEDLY SHARE PROJECTS WITH US EVERY WEEK

By the end of summer, it was clear that schools in a lot of countries worldwide were preparing to reopen while adapting to safety measures. This prompted us to consider how best to support educators; even though young people would be shifting back into a school routine, we still wanted to provide something they could engage with in their own time that would complement their computer science education.

We decided that the live stream was the best activity to continue engagement and made it an evening activity. Thus, Digital Making at Home pivoted to being a weekly live stream for young makers. To make it more accessible, we now stream to YouTube, Facebook, Twitter, and Twitch using a web-based broadcasting platform called StreamYard. We get the most engagement from Facebook, but our YouTube audience is quickly growing as well. Young people, parents, and educators can join our weekly live stream every Wednesday at 7pm GMT, 2pm EST, and 11am PST — and access our growing library of recordings at **rpf.io/home.**

## What's next for Digital Making at Home?

Nobody could foresee the changes that we would have to undergo globally in order to respond to the coronavirus pandemic, but we're all adapting, and we're still learning. The way that education is delivered has changed drastically, and we're now aware that this may have a lasting impact on the way young people learn. Our hope is that with Digital Making at Home, young people are having fun and learning computing at the same time. (HW)




### KEVIN JOHNSON

Kevin is the club programs coordinator for the Raspberry Pi Foundation North America team, working to support programmes like Code Club, CoderDojo, and Coolest Projects. He is also a Raspberry Pi Certified Educator. Kevin loves writing fictional short stories in his spare time and is currently exploring storytelling through photography (**@kjberrypi**).


## FIVE TIPS FOR CURATING ONLINE DIGITAL MAKING EXPERIENCES

- Focus on a theme for your session and structure your content around it. You may like to link your theme with something else your students are working on, to encourage cross-curricular learning.
- Make use of existing content for your sessions so that you are not reinventing the wheel. You can access a wealth of project resources at **projects.raspberrypi.org**.
- Encourage parents and siblings to get involved in your session too - this can be a really nice bonding experience for your students and their families!
- Invite special guests to your session to give a talk or run a workshop. They could be local tech professionals, volunteers, or even older children with an interesting project to share.
- Provide opportunities for students to showcase what they have created: showcasing should be fun and optional.

A young attendee of the online festival

# THE ADA SCOTLAND FESTIVAL: FOCUSED ON GENDER BALANCE

The festival launched on Ada Lovelace Day in October 2020 to inspire girls and women to explore the possibilities of a future in computer science

**The Scottish Qualifications Authority (SQA) introduced computer science into the Scottish secondary curriculum in 1980. In its first year, 2,432 girls signed up, representing 35 percent of the total.**

By the year 2000, the number of girls studying computer science had reached 9,031 — a modest increase over 20 years, and still only 35 percent of the total. By 2019, however, this figure had plummeted to 1,904 — fewer than the year of the course's inception. This compares to 8,282 boys for the same year, with the girls representing just 19 percent of the total.

If that is not discouraging enough, the low number of girls who actually progress through the three national qualifications at school level also gives rise for concern.

It is little wonder that higher education and industry are unified in their dismay over the low number of girls and women across all aspects of computer science education and in careers in tech. To identify ways in which the continuing gender imbalance in computer science might be addressed, Matthew Barr held a workshop bringing together employers, lecturers, and teachers from around Scotland. With the support of the Scottish Informatics and Computer Science Alliance (SICSA), three routes to improve gender balance were proposed:

- Addressing the challenges of bringing computer science into early years learning through engagement with teachers and parents
- Rebranding computer science to address misconceptions about the subject
- Creating a festival that brings together existing networks and organisations involved in improving gender balance in computer science education

Of course, gender is not the only diversity issue in computer science, nor is Scotland alone in this challenge; gender is simply where the SICSA group decided to make a start. Those attending the workshop all recognised that there are already many brilliant organisations and individuals working to address the gender gap, and bringing computer science to the forefront of young people's agendas. Girl Geeks, CoderDojo, CodeYourFuture, SmartSTEMs, Inner Wings, Equate Scotland, and 30% Club are just a few examples. Despite all these initiatives, and many internal programmes running in schools, colleges, and universities, computer science continues to suffer from a lack of diversity where other sciences fare better. With this in mind, it was agreed the first action from the workshop would be to build an umbrella festival to bring together, support, and amplify all the existing initiatives.

## SPONSORS AND SUPPORTERS OF THE FESTIVAL

The Ada Scotland Festival ran with the generous support of organisations large and small: Morgan Stanley; SICSA; dressCode; Fridays for Futuring:- IT4U; Barclays; and VeryConnect.

## FIND OUT MORE

helloworld.cc/ada.scot
twitter.com/adascot
facebook.com/adascotfest
linkedin.com/company/adascotland

And so the Ada Scotland Festival was born. It was co-founded by Matthew Barr, with the support of an advisory board and two co-founders. One of the co-founders is Toni Scullion, who is a computer science teacher and founder of dressCode and Turing Testers. She is also a member of the Scottish Computing Education Committee. Speaking about some of the reasons why she got involved, Toni said: "The figures from the SQA illustrate that the gap in computer science in Scotland at secondary school is widening. This is a really concerning trend, and doing something to help address this issue is something I feel extremely passionate about."

activities for girls. Aligning the festival with Ada Lovelace Day gave us the deadline we needed to get moving."

There followed fast and furious activity across Microsoft Teams and Zoom as the co-founders secured sponsors and invited partners to run events. Somehow, a website was built during a lunch break. What we were all unprepared for was the huge level of enthusiasm the festival received from across education, industry, and government bodies. Professionals with already maxed-out schedules were finding time to invite speakers and put together exciting events.

Opened by Richard Lochhead, Minister for Further Education, Higher Education, and Science, the festival culminated in over 16 events delivered by twelve different organisations and initiatives. Speakers included Baroness Goudie; Inner Wings founder and chief executive of SUSE, Melissa Di Donato; and distinguished engineer at Barclays, Theresa McComisky.

In addition to the live online sessions (many now available at **helloworld.cc/ada.scot-events**), activities and competitions

> "COMPUTER SCIENCE CONTINUES TO SUFFER FROM A LACK OF DIVERSITY WHILE OTHER SCIENCES FARE BETTER

### What is the Ada Scotland Festival?

A mixture of events and activities for women in tech and computing, the festival launched on 13 October 2020 to coincide with Ada Lovelace Day, which is celebrated each year in honour of Ada Lovelace, who was one of the first computer programmers and a pioneering woman in STEM.

One of the other co-founders is Ella Taylor-Smith, who is a senior research fellow at the School of Computing in Edinburgh Napier University. Ella had already been hosting events on Ada Lovelace Day at Edinburgh Napier University for several years, and was keen to expand the programme: "This year, I knew that we would need to move any event online. Matthew Barr was hosting a workshop on this theme and the idea came for an online festival that would unite the people working in this area, celebrate women in computing, and host events and created opportunities for participants to flex their tech and creative skills, consider their leadership qualities, and explore future pathways in tech. With the fantastic support received from businesses, education, and government bodies, and the excellent take-up we saw at the online events, the festival met its challenge to engage and inform. We hope it has helped girls and women to see the huge variety of pathways in tech and computer science and understand the importance of their role in these areas.

### What does 2021 hold?

The 2020 festival got off to a great start, and we've already started planning for 2021. In the meantime, we'll continue to support and promote events and initiatives across our media channels. The co-founders, along with everyone else involved in the festival, continue to strive for gender balance in their fields. The outstanding achievements of Toni's dressCode, in particular, are worthy of applause.

We have seen over the last year how crucial tech and digital connectivity has become to our work, social lives, and well-being. If we acknowledge how ubiquitous this has become, then we must also acknowledge that success cannot be achieved without a diverse workforce delivering technical solutions fit for all members of our community. In the words of Toni Scullion: "A more diverse workforce increases creativity, productivity, and efficiency. Diverse teams with a range of skills and perspectives innovate and solve problems better." (HW)

### MATTHEW BARR

Matt is the programme director for the Software Engineering Graduate Apprenticeship at the University of Glasgow. He is a co-founder of the Ada Scotland Festival, vice chair of the British Digital Games Research Association (DiGRA), and a director for the Scottish Game Developers Association (**@hatii_matt**).

### ANNA DOYLE

Anna is the liaison officer for the Software Engineering Graduate Apprenticeship at the University of Glasgow. She is a board member for the Ada Scotland Festival and a volunteer at CodeYourFuture. Anna also sits on the BIMA Scotland Council (**@UofGGAs**).

# CAN WE ENCOURAGE A GREATER UPTAKE OF GCSE COMPUTER SCIENCE?

The number of pupils — particularly girls — taking computer science GCSEs remains much lower than many other subjects. **Eleanor Overland** wonders if the options process itself is the key to unlocking change

**It can be very difficult for youngsters to make informed choices about the subjects they would like to study at GCSE level. GCSEs are the qualifications undertaken by 14- to 16-year-olds in schools and colleges in England, Wales, and Northern Ireland. These choices are made even more difficult by some schools offering a reduced period for Key Stage 3 (ages 11–14), and others offering computing as a carousel subject, whereby students only study the subject for a few weeks each year.**

Many computing teachers would like to increase the numbers of pupils, particularly girls, choosing computer science as an option. Currently, however, the options process — the process by which students choose subject options for their GCSE exams — is harder than ever to plan.

This article explores research in computing options and highlights potential actions schools could take in order to increase future numbers. While the research discussed in this article is based on findings from the UK, teachers in other countries will likely find similarities with their own students and options processes; and while the recommendations at the end of the article draw on UK examples, they may be broadly applied.

Decisions about future studies and careers can be daunting

## What the research found

The research on computing options explored in this article builds on the important analysis carried out in the Roehampton report (**helloworld.cc/roehampton**). This report is published each year by the University of Roehampton and examines the number of students who achieve GCSE and A level qualifications in computing. The headlines indicate that approximately 16 percent of eligible Key Stage 4 students (ages 14–16) take computer science GCSEs. Only 22 percent of this cohort are girls. Additionally, there is under-representation of students from poorer backgrounds and some BAME groups, especially Black students. For more on this report, see the Raspberry Pi research seminar presented by Dr Peter Kemp and Dr Billy Wong, available at **helloworld.cc/kemp-wong**.

To explore some of the stories behind the statistics, Professor Cathy Lewin and I did some research with pupils and teachers. This research took place prior to

the coronavirus pandemic, and we visited two contrasting schools in North West England. We surveyed all pupils after they had taken their options and interviewed a selection — including both those who had and had not opted to take computer science. We also interviewed the heads of department and teachers, and explored the options information materials. The pupils also completed mind maps and drawing activities to explore their wider perceptions of computing.

Both schools had secured healthy numbers of pupils for their GCSE computer science classes, but would have liked to recruit more. In line with the national trend, both schools had far more boys than girls choosing the subject. It is not possible to present the full analysis here, but I will summarise two key themes that emerged from the data.

## The influence of authoritative voices

Pupils were highly influenced by advice and by the experience of those around them, particularly older siblings or other family members. They were less influenced by their friends than teachers expected. Teachers also contributed to their decisions, with one pupil explaining that she chose computer science because her teacher told her she was good at it, and another citing a "persuasive assembly by the head of department".

> "MANY STUDENTS HAD ALREADY IDENTIFIED THEIR PREFERRED CAREER AND BELIEVED THAT COMPUTING WAS NOT RELEVANT TO IT

Career plans played an important part in the decision-making process. Many students had already identified their preferred career and believed that computing was not relevant to the path they had chosen. One girl planned to set up her own fashion company, but intended to employ others to do the technology side of things.

## The positioning of computer science

Some pupils found that the option blocks prevented them from selecting computer science, because it clashed with other favourite subjects. "I really want to do art, but really want to do computer science … it was really difficult, and stressful as well." Others talked about English Baccalaureate (EBacc) requirements, and how computer science did not contribute towards meeting these, although it was in the same option block as other EBacc subjects.

One school focused especially on pupils in higher-ability mathematics sets, promoting the subject more proactively to this cohort, although they did not prevent other students from taking up the course. The survey in this school identified a high number of pupils who did not take computer science as they felt it would be too difficult for them.

Gender perceptions were particularly complex. Many pupils described computer science as a subject open to both boys and girls, but also said that a lot more boys than girls chose it as a subject. The majority explained this by the factors already discussed. Some did comment that subjects are gendered — girls study dance and drama, boys study mathematics and science. Both schools are highly multicultural, and some pupils explained that gender expectations differed in some cultures and families.

Some pupils linked gender discussions to ability, one explaining: "Computer scientists

are very smart, or they could be an average person who knows their stuff and is into it." She explained that boys are more likely to play computer games and be "into computers", so could take computer science as an option even if they were "average", whereas the girls needed to be "very smart".

## Recommendations

This research focused on only two schools, yet revealed the similarities and differences between them. Teachers may wish to consider some of the following recommendations for their own schools when encouraging uptake of computer science as a subject:

- A short online survey can be useful to find out why current Key Stage 4 pupils have and haven't opted to take a computer science GCSE. It may reveal reasons specific to a school setting, perhaps different to those given above. Teachers can then plan specific actions to support their own pupils.

## ONLINE IDEAS TO SUPPORT OPTIONS

- Host exciting computing competitions or projects to get pupils involved in the subject before they choose their options
- Ask current GCSE students and alumni to make videos about what they enjoy in their computer science course
- Create an online gallery, such as a SlideShare, of GCSE classwork and projects
- Create a presentation targeted at parents and carers to provide information about computer science as a subject, further study, and careers
- Provide links to computing careers websites, especially those that promote a broader representation of people
- Ask local employers and higher education providers to tell pupils why computer science is important and what opportunities are available: perhaps you could put together a panel and host an online computing careers event
- Share your ideas with your local CAS community so they can make use of them too

- It's important to ensure that Key Stage 3 is memorable, particularly if a setting provides computing as a carousel subject, or relies on non-specialist staff to deliver at Key Stage 3. It may be useful to review schemes of learning, particularly prior to the options process, and ensure that these lessons are engaging and link to computer science at GCSE.
- Giving pupils self-belief can make a big difference. Is a GCSE in computer science open to pupils of all abilities in your school? If teachers appear to be selective and emphasise how difficult computer science is as a subject, they could deter other pupils who are capable of success. There are lots of resources available to encourage resilience and self-belief in computing which may be useful to explore, especially when it comes to supporting girls.
- It can be useful for schools to review their options structures. Are pupils being forced to choose between computer science and another of their favourite subjects? Are pupils following the EBacc requirements unable to opt for computer science? Perhaps just moving subjects into different option blocks could increase pupil numbers. This is something teachers may want to look at and discuss with their leadership team.
- Making careers education a regular feature of Key Stage 3 computing will help pupils to see the relevance of the subject in their future. Regular signposting to computing jobs, and highlighting the contribution of computing to other professions, can support pupil understanding. Display materials, guest speakers from industry, alumni, and parents working in computing can all really help with this.
- Teachers can share their successes. If a teacher already does something that effectively recruits pupils to take a GCSE in computer science, I encourage them to share it at a local Computing at School (CAS) community network meeting so that other schools can use the ideas. (HW)


### ELEANOR OVERLAND

Ellie is a senior lecturer in computing education at Manchester Metropolitan University. She is also a CAS Community Leader for Manchester Central, a CEOP Ambassador, and a Raspberry Pi Certified Educator (**@EllieMMU**).

# LOGGING ON AND BEYOND

**Sway Grantham** shares tips on how to teach children under seven to log on to school computers independently

**Just like blowing their noses or tying shoelaces, logging on to school computers is a skill young primary school students need to learn. However, many children can find logging on independently quite challenging. Here are some top tips to help you get your youngest learners logging on to devices independently.**

## Simple usernames and passwords

Consider what is making it hard for the learners to log on. If it's unrealistic usernames and passwords, then you should speak to the IT manager, or whoever controls your logins, and make these appropriate for the age and ability of the children. Their first name and a '1' for Year 1, and a three-letter password, is more than enough for this age group. However, you should avoid them all having the same password, as we do not want to advocate this, even with our youngest learners.

## Check and build the foundations

There is a range of foundational skills that children need before they can log on independently, including turning on and shutting down computers safely, using a keyboard, and using a mouse to click in a box. Do your students have these skills already? If they don't, it's worth spending some time on developing them, so that logging on isn't an overwhelming task.

## Teach it explicitly

If you plan a lesson in which the outcome of the lesson is that learners have logged on successfully, you can break down each step and take your time, without feeling the pressure of moving on to other lesson content. For those learners who still struggle, a visual prompt is often useful — you can hand them a chart showing each of the steps, to encourage them to continue independently.

## Peer support

Can the children who have mastered the skill support those who haven't, until they become more independent? I would often set table challenges for everyone on one table to get logged on, awarding school points for the fastest table. This encouraged the more capable children to support those who were struggling, reducing the time pull on the teacher, and also allowing the learners to learn from their peers. I also had a rule that the only person allowed to touch the computer was the person who was using it, so the helpers could tell them what to do, but not do it for them!

## Don't let it be a barrier

If, after trying all of the above, you still have some learners who are struggling, they should still be able to access the computing curriculum with support. Just like if a child was unable to dress themselves for PE, you would eventually intervene, but you might set them a personal target to work on that specific skill until they got there. We don't want to impact the children's attitude to computing because they struggle with logging on.

Logging on is a skill that many adults take for granted, but it is a skill in itself. We need to give it the amount of time it deserves, so that we can get the majority of our learners to do it independently as soon as possible. This not only makes our lives easier, but it also broadens the learners' opportunities for using technology across the curriculum in the future. (HW)

SWAY GRANTHAM

Sway is a senior learning manager at the Raspberry Pi Foundation, where she leads a team developing computing resources for primary teachers (**@SwayGrantham**).

# FINDING FLOW IN THE LEARNING PROCESS

**Francisco Mireles** explores the theory behind the state of flow and shares how he applies this knowledge with his students using the concept of low floors, high ceilings, and wide walls

**It's 3 am and you are absorbed in your favourite video game. You are overcoming challenges and winning at every turn. You are fully present — so much so that you don't even feel time passing by, because you are engaged and enjoying every second.**

The state of consciousness described above is better known as the flow state. As defined by renowned psychology professor Mihaly Csikszentmihalyi in his book, *Flow: The Psychology of Optimal Experience*, the flow state is "the state in which people are so involved in an activity that nothing else seems to matter; the experience itself is so enjoyable that people will do it even at great cost, for the sheer sake of doing it". People may also enter a flow state when they are engaged in things such as playing music, creating art, or playing sport — essentially, any endeavour that they can become fully immersed in.

Csikszentmihalyi further describes how you reach this consciousness sweet spot, including several conditions needed to achieve it. These conditions include: that the activity you are performing demands enough of you to avoid boredom and keep the experience challenging — but is not so challenging that it could make you anxious; that the efforts you put into the activity are voluntarily given; and that you find the activity worthwhile.

It can be said that the more you engage with a state of flow, the more optimal experiences you may have. The more optimal experiences you have, the more you stretch your skills and develop your knowledge. I would guess that many people who have come across the flow concept have seen it as a way of improving productivity. I have found that there is a bigger, more profound benefit to achieving flow in any kind of activity —

FRANCISCO MIRELES

Francisco has a background in industrial design and software design, and is currently a professor at the Digital Planet Lab at Docet School in Monterrey, Mexico, where he teaches digital skills to elementary students. Francisco has also given workshops about education, prototyping, and creativity (**@frank_mireles7**).

Teachers set their students' learning in motion as they would set rocks rolling down a hill

that you become happier. The more you lose yourself in optimal experiences, the more you take control of what is happening, which can in turn be very satisfying.

Now, wouldn't it be great to apply what we know of the flow state to the way we, as educators, design learning experiences? In this article I will explore how teachers can encourage a flow state in their students through the concept of 'low floors, high ceilings, and wide walls' as described by the renowned educator Seymour Papert and MIT's Mitchel Resnick. I will also share how I put this concept into practice with my fourth grade technology class.

## Applying the flow state concept in your classroom

If you are a teacher like me, you probably want — as much as I do — to stretch all of your students' cognitive and creative skills to the maximum and have them enjoy every second they are at school. But you also know how different they are; every student has their own particular set of skills, knowledge, and interests.

### Low floors, high ceilings, and wide walls

When thinking about how to apply the flow state concept in my own classroom, I had the following questions:

- How can I provide the specific challenge needed for each student?
- What should I decide to teach, when the interests of my students are so varied?

> "WE ENTER A FLOW STATE WHEN WE ARE ENGAGED IN ANY ENDEAVOUR IN WHICH WE CAN IMMERSE OURSELVES FULLY

Other teachers looking to encourage a flow state in their students will probably have similar concerns. I decided that these two questions can be addressed with the concept of low floors, high ceilings, and wide walls, brilliantly described by Mitchel Resnick, director of the Lifelong Kindergarten Research Group in the Media Lab at MIT, in his book *Lifelong Kindergarten: Cultivating Creativity Through Projects, Passion, Peers, and Play.*

As for the first question, Resnick writes: "Seymour Papert often emphasized the importance of 'low floors' and 'high ceilings'. For a technology to be effective, he said, it should provide easy ways for novices to get started (low floors) but also ways for them to work on increasingly sophisticated projects over time (high ceilings)."

To use an analogy: I see a teacher as someone who tries to get all of their rocks down a hill by setting them in a rolling motion — the rocks being the students in this scenario. In any given lesson, some students get off to a slow start, while others race ahead. It is our job, as teachers, to set them in motion so that they can roll on their own, while still providing support if they get stuck. Sometimes the ones that were behind suddenly exceed every other rock, while the ones that were at the front may fall behind.

By designing activities that have low floors, we can ensure that every student can get started, no matter what their experience. Similarly, by making sure that the activity has a high ceiling, we ensure that students who have already achieved the minimum required, and want to increase the complexity, can go for it.

## FURTHER READING


- Csikszentmihalyi, M. (1990). *Flow: The Psychology of Optimal Experience.* [e-book] New York: HarperCollins, p.4
- Resnick, M. (2017). *Lifelong Kindergarten: Cultivating Creativity Through Projects, Passion, Peers, and Play.* Cambridge, Massachusetts: The MIT Press, p.64

As for the second question, regarding your students' wide range of interests, Resnick writes: "As my Lifelong Kindergarten group develops new technologies … we add another dimension: wide walls. That is, we try to design technologies that support and suggest a wide range of different types of projects."

To return to the rolling stones analogy, what I aim for as an educator is for every rock to roll through its preferred path; some of them may prefer grassy surfaces, and others, sandy paths. The idea behind this part of the analogy is that students should be able to pursue their interests through the activity itself.

## An algorithmic example

In my fourth grade technology class (ages 9–10) we are working on a project called Algorithmic Art Sticker. The idea is that every student creates a cool piece of procedural art that then gets printed on sticker paper and sent to every student's home. We are using the Artist tool from **code.org**.

The results so far have been astounding. To refer back to Papert's and Resnick's ideas about low floors, high ceilings, and wide walls, as discussed earlier, here are my thoughts as to why this may be:

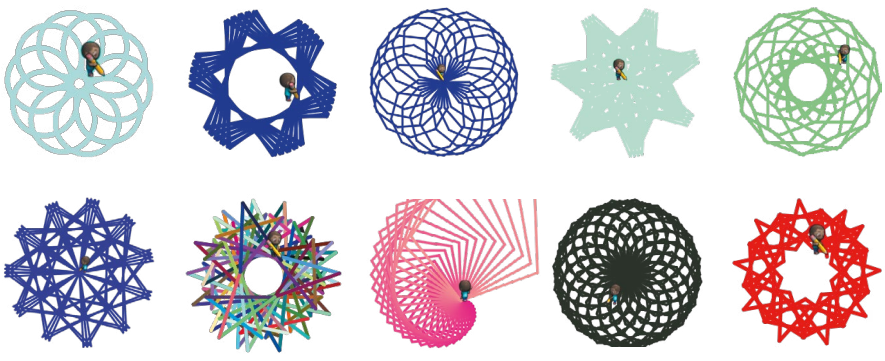
- **Low floors:** During each session, students learn a new concept that sets the standard of what must be done as a minimum.
- **High ceilings:** As students only have one session per week, at the end of each session they are encouraged to keep playing and experimenting, in order for them to understand what other things could be done.

> "ACHIEVING FLOW, AS WELL AS INCREASING YOUR PRODUCTIVITY, HAS THE PROFOUND BENEFIT OF MAKING YOU HAPPIER

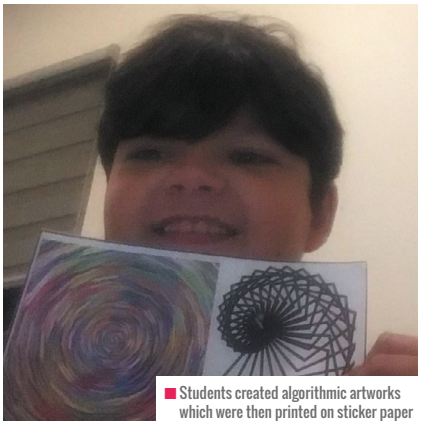
- **Wide walls:** Students are invited to try different shapes, colours, and even themes for their work. The aim is not just to get their creative engines started — which is definitely important — but also for them to make their projects their own. In my experience, students become more invested when they are working towards something that they have defined themselves.

While looking through my students' projects, I noticed something nice — that there were many versions of the projects. It's worth mentioning that while I asked them to make one project to work on during our sessions, I also told them that they could experiment freely, without worrying about messing anything up that would impact on future sessions. They were free to create other projects to play around with — and they did; some students even had five or more projects! That proved to me that some of my students achieved a flow state; they strove to put the concepts into practice and to advance their existing skills — and for some, this resulted in multiple projects.

So here is my invitation: let's work together as educators towards getting all of our students rolling within the parameters of the flow channel. By designing learning experiences with low floors, high ceilings, and wide walls, we can support our students' engagement and growth. (HW)



Examples of the students' algorithmic art creations



Students created algorithmic artworks which were then printed on sticker paper

# A JOURNEY INTO PHYSICAL COMPUTING

**Rebecca Franks** shares some of the barriers to introducing physical computing into the classroom, and how you can overcome them

**During my 15 years as a computing teacher, I always wanted to learn more about physical computing and develop activities for the classroom, but I had many barriers in my way. Now that I'm out of the classroom and developing resources for the computing curriculum in the UK, I have learnt how to remove those barriers. Physical computing is really inspiring, but doesn't always appear on a typical computing curriculum.**

By exploring physical computing, I have found out why it is so important that young people experience it. I have spoken to educators around the globe who are using it successfully, and discovered some of the challenges that teachers face when trying to implement it in their classrooms. I have found some fantastic resources and learnt some new skills, which I will share with you in this article.

## Why physical computing?

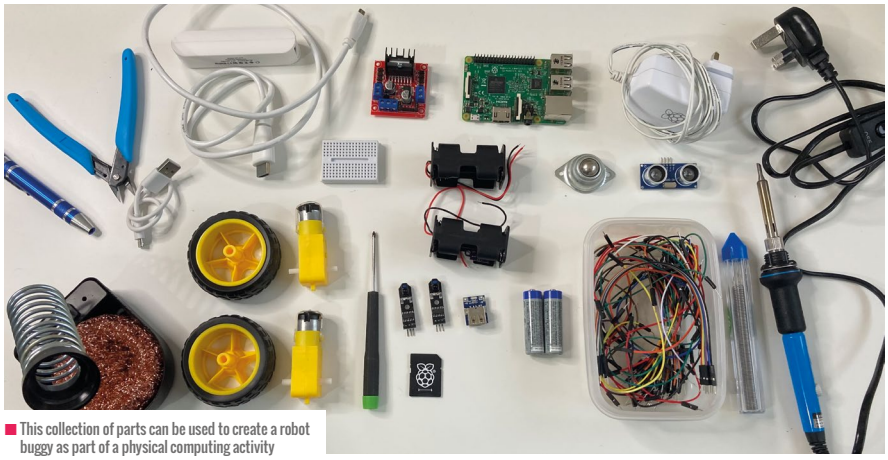
From making an LED blink, to programming a robotic turtle such as a Sphero, or creating a weather station, physical computing takes programming away from being solely on the screen and moves it into the real world. When a student's project works, they can see their creation move, or light up, or record some data from the environment in real time — and these can be brilliant 'Aha!' moments.

Recent research has highlighted that physical computing can play a huge role in increasing the motivation of learners, and in particular, the motivation of girls. Learners are creating real, tangible devices that give them instant feedback.

It's been found that girls tend to like creating things that make a difference to people's lives. If you set a class a project in which they need to make a device that can help with wider issues, like global warming, this can really motivate them to learn more about how to program and build their own devices. Physical computing also lends itself to an interactive and collaborative approach in the classroom, which is an important factor in girls' attitudes towards the subject, too.

A lot of the research on physical computing education has resonated with my own experiences. As a child in IT lessons, I loved drawing patterns with Logo on the BBC Micro, and became quite proficient at using the applications on the Acorn Archimedes. I also really enjoyed my DT lessons, in which we would use the machinery to make clocks out of acrylic, and solder wires and batteries to make pinball machines. At that age, I would never have thought that computing and soldering were linked in any way. It never really came up. And I had no idea that you could have

> PHYSICAL COMPUTING INCREASES MOTIVATION AS LEARNERS CREATE TANGIBLE DEVICES THAT GIVE INSTANT FEEDBACK



This collection of parts can be used to create a robot buggy as part of a physical computing activity

Ali Alzubaidy leads a lesson using micro:bits with his Code Club in Iraq

> THEY LEARN WITHOUT REALISING IT, AND I'VE FOUND THAT THIS IS THE KEY TO CREATING POWERFUL, ENGAGING LESSONS

a career that used those skills. It was a classic situation of, 'I couldn't see it, so I couldn't be it.'

This is why I believe it is so important that we introduce physical computing in our schools. It opens doors to all sorts of opportunities that learners might not otherwise be aware of.

## Physical computing in clubs

The cross-curricular and project-based nature of physical computing, in which children can work on designing, programming, and making a product with an end goal, lends itself well to being a part of informal learning settings such as coding clubs.

Gary Quinn, who runs a Code Club at Lostock High School in Greater Manchester, uses a range of physical computing devices with his students. He uses the Makey Makey — an integrated input/output device that allows students to create devices such as keyboards made of fruit — and the micro:bit, a microcontroller board which he uses with electronics kits. He told me: "Nothing beats that feeling of achievement students get from seeing their code run before their eyes in the real world. They love to show off their creations to peers and staff alike! The students always comment on the fun factor and want to see every project through to a conclusion. While they are learning about code, circuits, accelerometers, and everything in between, it doesn't feel like an effort to them."

Quinn values the engaging impact that physical computing has on his learners: "The excitement and anticipation are unmatched by paper or screen-based activities, and gone are the days when Scratch the cat moving across the screen would suffice."

Ali Alzubaidy, a Code Club volunteer in Iraq, uses physical computing to demonstrate to his students the links between technology and other subjects. For example, to teach his group about the geography of Iraq, he loads a map of the country into Scratch, and learners create the code to allow a Makey Makey device to interact with it.

He describes the delight in his class when working with physical computing: "When the kids made anything with tech, they were so happy." Ali told me he believes that physical computing is important because it builds students' confidence with technology,


Rebecca's latest make is a postcard with decorative LEDs attached

improves their logical thinking, builds team-working skills, and prepares students for their future careers.

## Challenges of getting physical computing in the classroom

So if physical computing is so popular with learners, and it has positive effects on their learning, why isn't there more of it in computing classrooms?

For me, the limiting factor was time. When I was a teacher, I purchased an Arduino starter kit — a small microcontroller board with modular sensors and electronics — to see if I could start building my own projects. I had success following the set tutorials, but really struggled if I wanted to make something independently. I kept trying to get into physical computing, but I found it tricky to put time aside to dedicate to really understanding how it all worked.

Research has shown I'm not alone. In 2018, the Raspberry Pi Foundation published a report that found Raspberry Pi computers — single-board computers that lend themselves to physical computing projects — are quite popular in schools, but that teachers often stick to simple projects that rely on step-by-step instructions, rather than giving learners more open-ended tasks. This is potentially due to time constraints, but also the knowledge and experience of the teacher. An open-ended idea can lead to the teacher needing a wide range of physical computing knowledge.

For teachers to start using physical computing in their computing lessons, they need access to high-quality lesson plans. These resources can help to build teachers' confidence, making them more likely to use

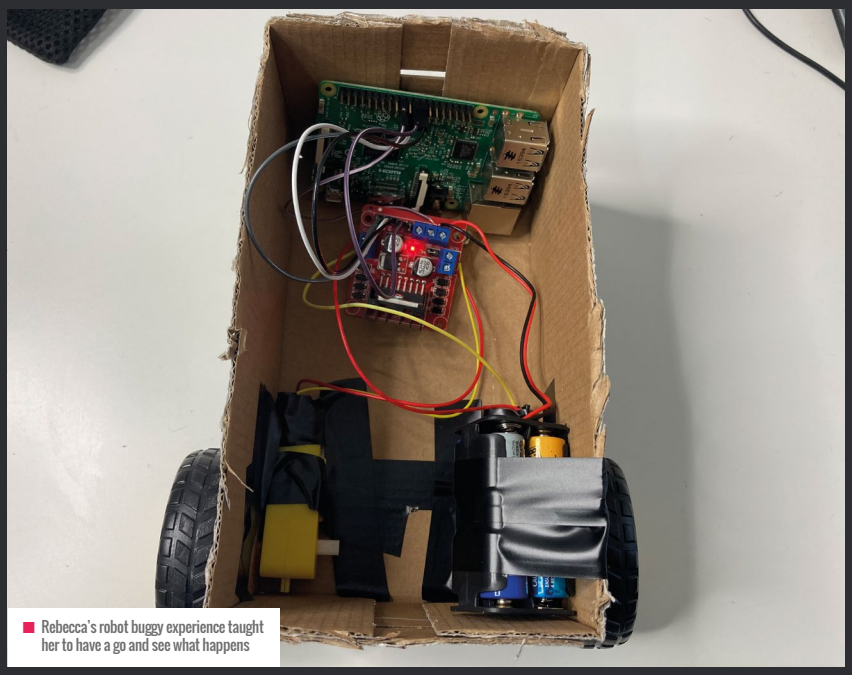
## MY ADVENTURES IN PHYSICAL COMPUTING

"If I cut this wire and solder it to another wire, will it still work?"

This is a question that I found myself asking on my work Slack channel a few months ago. At the time, I was working through the Raspberry Pi 'Build a robot buggy' course (**helloworld.cc/buggy**). Instead of just soldering some wires together and seeing what happened, I went straight to my colleagues and asked the question. I wondered, "Why am I asking this question, when I could just solder the wires together and see what happens? Why don't I have the confidence to just do it and see what happens?" I was so apprehensive about simply jumping in and trying something.

It can be really intimidating to be around people who know so much more about a subject than you. Low self-efficacy is quite a common thing for women to experience, and I definitely fell into this category when it came to physical computing. I had a little chat with myself and decided that, from then on, I was just going to give it a go and see what happens. This change in attitude has been liberating for me.

I made a commitment to just have a go at things and stop questioning myself. I started shoving loads of bits of card into my buggy to make the wheels work a bit better. I unscrewed and swapped out the wheel bearing, as it had gone sticky, and I replaced a broken sensor, and tried different ways to stick parts into the box. After lots of trial and error, I had a fully functional robot buggy, and it felt great!



Rebecca's robot buggy experience taught her to have a go and see what happens

Since building my buggy and learning how to solder properly, I have become much more confident at trying new things. And yes, I did solder those jumper wires together, and yes, they did work! I think as adults, we can sometimes feel as if we are wasting time if we just sit and have a go at making something. We feel like it has to have a purpose or an end goal. It really doesn't; you can get so much satisfaction from making a random thing. My latest make is a postcard with decorative LEDs.

physical computing in their lessons.

The Teach Computing Curriculum (**helloworld.cc/tcc**) provides comprehensive physical computing units. They demonstrate clearly how to progress learners' understanding of physical computing devices such as Crumbles and micro:bits. Lessons include guidance for the teacher and helpful guides that learners can use to help them debug any issues that they may encounter. The resources are published under an Open Government Licence and are free to download from anywhere in the world.

### Getting started

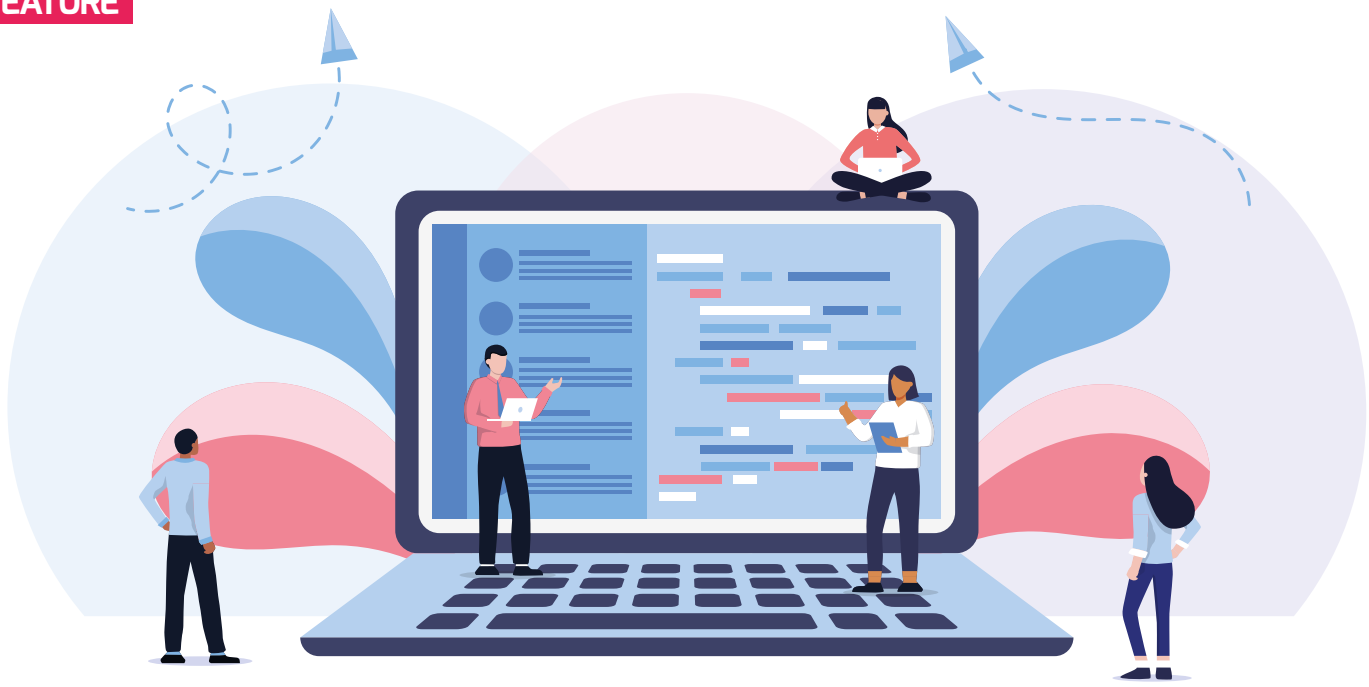
Physical computing is so important for our young people. It helps them build connections between the real world and programming, while giving them something exciting to focus on. They learn without realising it, and I've found this is the key to creating powerful and engaging lessons.

If you are in the same position that I was in with physical computing — wanting to start, but feeling nervous or uncertain of how to do so — then my advice is to just relax and enjoy making something. Buy some really cheap components and have a go. See it as a journey, not a race! And try not to compare yourself too much to some of the awesome makers out there. They were all in the same place you were once, and will be very willing to help you out if you need some advice. You won't regret it — and you never know where your adventures could take you! (HW)


**REBECCA FRANKS**

Rebecca is a learning manager at the Raspberry Pi Foundation. She writes resources for the Teach Computing Curriculum. She taught computing for over 15 years (**@FranksberryPi**).

FEATURE



# GETTING EVERYONE EXCITED ABOUT CODE

**Vanessa Bakker** shares how she overcame her hesitations and, in response to local interest, started a friendly and informal programming club for adults

**I am very motivated to tell people — both children and adults — that coding is fun. I had already been running a CoderDojo club (for young people aged 7–17) at the public library in Almere for over a year when I approached the programme manager there about starting a friendly and informal coding club for adults. This came about after many of the parents at the CoderDojo asked me where they could learn programming skills. So I spoke with the library and they felt it was a good idea to try it. They were interested to see whether it was something that people would sign up for. By the second day, we were fully booked and we had 40 people on our waiting list — and so we realised that it was kind of a success!**

I started the adults' coding club in September 2020 with Florus van der Rhee, who is also a volunteer at my CoderDojo. We planned to run ten sessions in total: five on Scratch and five on Python. We held the first two sessions at the library, but have held all the following sessions online due to coronavirus restrictions. Unlike with CoderDojo, we charged participants a small fee for the block of sessions — not to make a profit, but to inspire a sense of commitment from the participants.

## Diverse backgrounds

We have 15 participants at the club. The youngest one is 18, and the oldest is 76 — a gentleman who wants to keep his mind fresh. They are all from different backgrounds: we have full-time parents, students, people involved in social and community work, and even a yoga instructor. Initially, I was very curious about what attracted everyone to the club. Here in the Netherlands, we have the word *gezellig*. In English, it can be understood as 'cosy'. And that's what our club is like: it's a place to get to know programming in a friendly environment. It's not competitive — there are no exams — and the focus is on peer-to-peer learning and becoming comfortable with code.

I was also pleasantly surprised by the number of women who signed up to our club — twelve out of our 15 participants are women. Unfortunately, I think the perception is still out there that coding isn't for girls. We still have a lot of work to do in getting girls and women to believe that programming is for everybody. When I asked the women about what attracted them to the club, they said that part of the reason was that there was a woman running the club — my face was on the advertisement! So it was kind of a role model thing, whereby they thought, "If she can do it, then I can do it." So we have to make sure that there will be more role models to help women to think like that. They also liked the fact that the club had a low profile. It doesn't matter if it doesn't work out; the point is to come along, learn, and have fun. And that really helped women to get over their hesitation.

The coding club for adults at Almere public library

> ONE OF THE MAIN THINGS I'VE LEARNT IS THAT YOU DON'T HAVE TO BE A TOP CODER TO GET INVOLVED IN SOMETHING LIKE THIS

## Challenging my apprehensions

While I was excited about starting the club for adults, I also had some apprehensions. Firstly, I am not a software engineer — I studied educational sciences at university. I was a little bit uncertain about working with the adult participants, because they want to do all of this creative stuff, and expect the club leader to be an expert in everything! They thought that because I was organising this coding club, I was a real coder, who studied computer science and works as a software engineer. In other words, someone who knows everything, which I don't! And that was kind of scary for me. You can start something and think: "Oh, I have to know everything and then I can start." But actually, I decided: "No, I'm just going to start." And I don't know everything.

My co-host, Florus, also reassured me that it's OK not to know everything right away. If someone asks a question that I don't know the answer to, I can tell them that I'll think about it and get back to them later. Real software engineers don't know everything right away, either!

## Teaching adults vs teaching children

I have been running the CoderDojo club for children for over a year. Some of the young people are 14 or 15 years old now, and have been attending for the whole year. And they say, "Well, yes, I love coding, and I really want to do something with coding after I graduate high school." That's part of why educators like me do it — to get people excited. Of course, nobody needs to go on to have a career in coding, but it is a bonus if they say they want to do it professionally. As long as they're excited, that's fine by me. Getting people excited about coding is something I was eager to achieve with my adult club participants, too.

I was a bit apprehensive about teaching adults, given that my experience up until now was mentoring children at my CoderDojo. As I mentioned previously, while I am confident coding in Scratch and Python, I don't know everything — and initially, my adult participants assumed I had all the answers. They had thousands of questions that the children at my CoderDojo wouldn't have. We used the free project resources from the Raspberry Pi Foundation, which have worked really well for us.

One of our participants told me that his teenage son had laughed when he told him we were coding in Scratch, because Scratch is mainly taught to children. I explained to the adults why I thought Scratch is a good language for learning the basics of coding; while it looks like a child-oriented environment, it is perfect for learning the basics, as many teachers will know! Also, to jump straight into Python in the beginning without having learnt Scratch would have been too big a step, in my opinion. Once I explained this to the adults, they were very happy to start with Scratch. They also learnt that some Scratch projects are quite advanced!

## Final thoughts

Whether you are starting a programming club for children or adults, it's important to recognise that you don't need to have all the answers — just start. I'm very happy that I began my club for adults, and I am learning so much from it, too. One of the main things I've learnt is that you don't have to be a top coder to get involved in something like this; being enthusiastic and having people skills are just as important as programming skills. I would also recommend surrounding yourself with people who want to learn, and who want to make a positive learning environment. And if you have those people, then you have a very good club. (HW)


### VANESSA BAKKER

Vanessa lives in Almere, the Netherlands, where she works at the city hall advising the local government on education and ICT. She runs a CoderDojo club for children, and a programming club for adults, from the local library. Vanessa is also the mother of two young boys.

# ARDUINO SCIENCE JOURNAL: A FREE SCIENCE LAB IN YOUR POCKET

Students can conduct scientific experiments and investigations on the go with this free app, explains Arduino's **Valentina Chinnici**

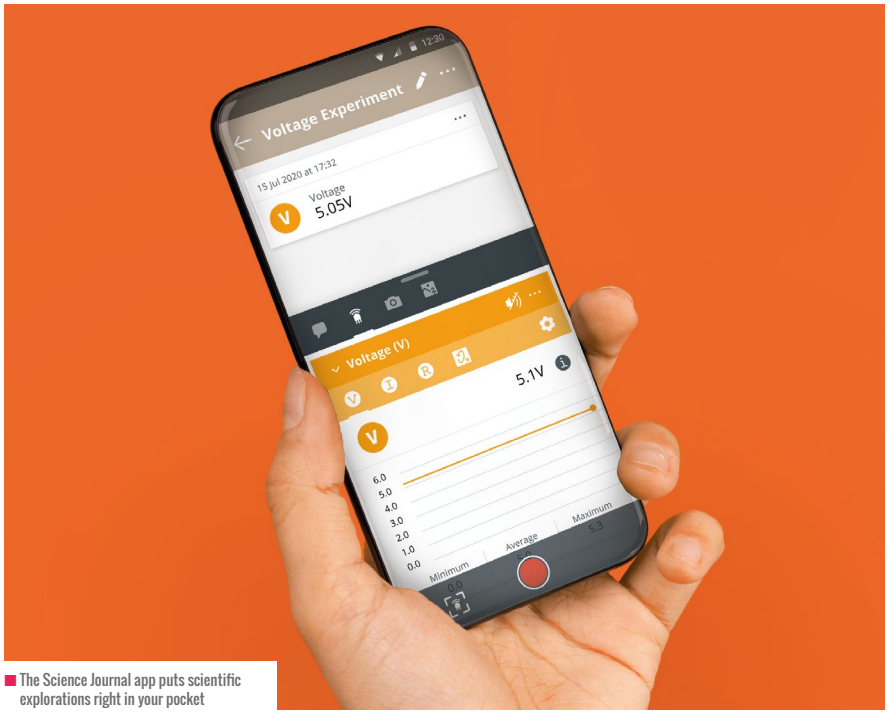
**For many students around the globe, accessibility to STEM education has been compromised recently; among other things, the pandemic has dramatically impacted students' access to scientific lab equipment. But what if a student's science lab could follow them anywhere, in their pocket? And what if it were possible for them to record and analyse data directly from their smartphones?**

These questions have been answered with the release of Arduino Science Journal — formerly known as Google Science Journal — a mobile application for scientific explorations.

Arduino has been enabling hands-on learning experiences since its foundation in 2005. With the Science Journal app, we have taken on the challenge of offering a new free tool to our large community of educators. With this tool, we aim to support them in teaching STEM remotely.

## What is the Arduino Science Journal?

The Arduino Science Journal is a free, open-source mobile app. It allows teachers and students alike to conduct scientific experiments by measuring the surrounding world with sensors; recording, documenting, and comparing data; developing and validating hypotheses; and taking pictures and notes. In other words, it is like a portable Swiss Army Knife for educational science experiments.




The Science Journal app puts scientific explorations right in your pocket

## Harnessing the power of sensors

The only thing that students need to run their experiments is a smartphone. Using the app, they can easily access the sensors in their mobile device and use them to measure the world around them. They can explore properties like light, movement, and sound, thanks to free tutorials that guide the learner in the assembly of fully operational data loggers. For example, a tutorial on the light instrument (**helloworld.cc/sj-light**) introduces learners to a musical instrument that plays notes according to the light conditions in a room. One student from Boulder High School in the USA said: "I like seeing the things I'm hearing." The app translates ambient light into sound, to support kinaesthetic learners. All the experiments can be recorded and played for later analysis. Angevine, a teacher of grades nine to twelve in the USA, shared: "It's really cool to take measurements from something in your pocket."

School budgets often don't have enough resources to buy newer lab equipment, especially when students need an item each. That's why it is so useful for students to be able to leverage the sensors available in their own devices. Teachers love the Science Journal because you don't need any equipment other than a smartphone. All the sensors are in one place, and you can collect accurate data in a few minutes, with the appropriate instant graphing, and without the risk of losing data. The idea of using phones meaningfully is also something that is popular with the teachers we have spoken to, particularly when it allows for exploration and enables an enquiry-based approach.

The fun doesn't end with students' smartphones, either. The app can be paired

Students use the Science Journal app to record and analyse experiments

with external sensors and hardware using Bluetooth to unlock a world of possibilities. The Science Journal is fully compatible with the Arduino Science Kit, the Arduino Nano 33 BLE Sense board, and other third-party hardware, giving students access to dozens of sensors. In addition is the Arduino Science Kit Physics Lab, which includes nine traditional physics experiments applied to amusement park rides. One of the best examples that combines physics and fun is the Pirate Ship experiment. Students can experiment with pendula, observe their momentum, and gather data to observe, while learning the history of the Pirate Ship ride and the theories behind it.

> “ EXPERIMENTS COVER AREAS SUCH AS LIGHT, SOUND, MOTION, AND ELECTRICITY

## Curriculum alignment

When developing products for educators, we strive to align our content with international STEM curricula. The Arduino Science Journal app is suitable for students aged ten and older, making it the perfect companion for learners in middle school through to upper school. The curriculum covered includes Key Stages 2–4.

At Arduino, we aim to make educators' lives easier, so that they can focus on the thing they do best: teaching. Detailed curriculum grids, available on our website at **sj.arduino.cc** (scroll to the bottom of the page for the downloads), map the functionalities of the app with the science programme of study, to simplify the assessment of students when running experiments using the app. The mapping can guide teachers when developing lessons based on the scientific method, and help them to make sense of how the app can be used in a classroom context. The app supports students in planning experiments, making observations, testing hypotheses, and exploring phenomena. It also helps students to translate data from one form to another. With the app, quantitative measurements, from sound frequency to ambient light, are made simple.

The app can be freely tailored by educators to a variety of learning situations, both in and out of the classroom. It is suitable for individual use, collaboration in small groups, remote learning, and homeschooling. The topics covered by the app include the understanding of the scientific method and how to apply it to scientific research, as well as science subjects in general such as chemistry and physics, with free activities available at **science-journal.arduino.cc**.

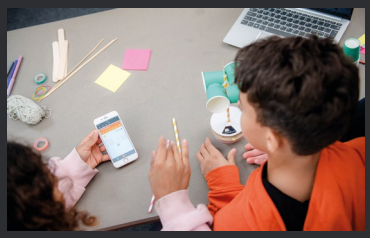
## Flexible teaching materials

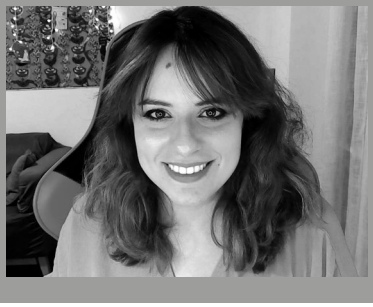
At Arduino, we are on a mission to make education more accessible — that's why the app is free for all users. Students can choose between dozens of hands-on experiments designed by education experts.

Experiments cover a range of areas, such as light, sound, motion, and electricity. The content can be used to enrich a student's learning experience in a variety of subjects such as maths, physics, biology, and chemistry. Our goal is to give teachers flexible teaching materials that can be easily tailored in their lessons. (HW)

## DATA LITERACY FOR ALL

**The Arduino Science Journal app allows your students to record, store, and export data, create graphs, take notes, and snap high-quality photos. This helps students to document and present their findings like professional scientists.**





### VALENTINA CHINNICI

Valentina is product manager of the Arduino Science Kit Physics Lab, co-developed in partnership with Google, as well as the Arduino Science Journal app. She has facilitated educational workshops worldwide at events such as FAB10 and Maker Faire Bay Area. In 2015 she won the IoT Postscapes award with her IoT birdhouse (**@sorciererouge**).

# RETRO VIDEO GAMES IN THE COMPUTING CURRICULUM

Hook or deterrent? **Neil Rickus** explores when, why, and how old video games could be used in computing classes

**As a child of the 1980s with a computer science degree, it's perhaps unsurprising that I love playing retro video games. And I've found that games such as Pong, Space Invaders, Pac-Man, and Super Mario Bros can enrich the curriculum and help children to develop their understanding of digital devices, while also providing an opportunity to examine the history of computing.**

But by using games in lessons, we could potentially be projecting wrong ideas about what computing is for: children might come away knowing that by studying computer science they could make games, but what about developing open-ended, collaborative, and creative projects, too? Also, gaming is often seen as being solely a male pursuit, so we could risk alienating a large portion of our learners. I'd like to explore the opportunities for incorporating games in the computing classroom while ensuring that all students feel there's a space for them.

NEIL RICKUS

Neil is a senior lecturer in computing education at the University of Hertfordshire, a primary education consultant for the BCS, and the founder of Computing Champions. He is a CAS community leader, a CEOP Ambassador, and a Raspberry Pi, Google, and Microsoft Certified Educator (**@computingchamps**).

## Curriculum links at different ages

For young learners up to around the age of five, retro video games could provide an opportunity to interact with technology in a safe environment, without having to worry about the online safety dangers posed by modern devices. For example, games such as Pong can be operated with a simple controller to move the on-screen paddle. This teaches children how they can interact with a machine, along with developing their hand–eye coordination. Other controllers, including joysticks, joypads, and keyboards, could provide further opportunities for children to use technology and develop their confidence with digital devices.

The English primary computing national curriculum (for ages 5–11 years) outlines how children should be able to produce programs containing sequences of instructions, along with using the concepts of selection, repetition, and variables. Modern block-based programming environments, such as Scratch, enable children to produce programs comparable in complexity and quality to retro video games. Games such as Space Invaders or Super Mario Bros could be used as a stimulus for learning, along with an opportunity to discuss how programming concepts have been implemented. For example, the movement of a typical enemy in Space Invaders might proceed in a manner similar to the instructions that follow, which could be implemented in Scratch:

```
Repeat until touching laser:
    Move 2 pixels across screen in
    current direction
    If touching edge of screen:
        Change direction of moving
        across the screen
        Move 2 pixels down the
        screen
```

Many retro video games are collaborative in nature, such as Bubble Bobble or Gauntlet, and players have to work together to defeat a common enemy. This can enable children to develop their communication and social skills while using technology. It might also help to improve children's resilience as they attempt to solve a problem or complete a level, or when they lose lives in a game, although care needs to be taken to balance the possible enjoyment of completing a section of the game against creating an overly competitive classroom environment.

As children enter secondary school (ages 11–16 years), curriculum content focuses more on how digital devices function, which includes the importance of binary. The graphics in retro video games, which were often constructed using a limited number of pixels, enable concepts such as bits, colour depth, and image size to be studied. Storage can also be examined, and comparisons made with modern systems.

Other teachers I've heard from have also shared how text-based adventure games, which were popular on early home computers, can be developed in Python by secondary students. These games enable subroutines, variables, and arrays to be implemented, and can even link to other subject areas, including English.

Interacting with vintage computers and video games consoles lets students see how technology has developed over the years

## Is it actually all about history?

The impact of technology might also be considered by students when studying computing. For example, in the US, the Computer Science Teachers' Association Standards have the following statement for students in grades three to five (ages 8–11): "Discuss computing technologies that have changed the world, and express how those technologies influence, and are influenced by, cultural practices."

Perhaps there is an opportunity for retro video games to become part of the computing curriculum in the UK? The huge leaps in processing power, storage capacity, and display technology during the later part of the twentieth century were phenomenal, and heavily influence how we use our digital devices today.

As the use of retro video games can enable children to interact with older home computers and consoles, there's also the opportunity to study the history of how hardware and software have developed. Machines such as the ZX Spectrum and the BBC Micro have heavily influenced modern computing, while bedroom coders — amateur programmers who worked by themselves — produced some of the most memorable titles in video game history, and forged the beginnings of the billion dollar gaming industry today.

Organisations such as the Code Show in the UK specialise in bringing original devices into schools for children to use, while museums, including Cambridge's Centre for Computing History and The National Museum of Computing at Bletchley Park, enable visitors to use an extensive range of machines. Modern systems, including the Raspberry Pi, can also be used to emulate older machines.

> "COLLABORATIVE GAMES CAN HELP TO BUILD RESILIENCE AND DEVELOP SOCIAL SKILLS

## Video games and computing

While video game use is increasing among all demographics, this doesn't necessarily translate into more young people choosing further study of computer science, and video games are regarded by some as an activity for teenage boys.

Research has outlined how boys' childhood use of computers through playing video games means they have playful experiences with technology, rather than seeing digital devices as tools. This can translate into boys feeling more comfortable when interacting with computers. Perhaps then, through providing opportunities for all learners to interact with simple retro video games in lessons, such as those produced for early computers and video games consoles, we could enable learners to increase their confidence with technology, while also developing their knowledge and understanding of the computing curriculum?

In my opinion, there could be a place for retro video games in the curriculum, if this is balanced by including other non-gaming applications of computer science — such as artificial intelligence, data science, and web design, in conjunction with collaborative, creative projects that have a positive contribution to the world around us. This would encourage learners to find areas that might interest them, as well as introducing the broad scope of careers available through the subject.

What are your thoughts on using retro video games in schools? Is their use suitable for the mainstream classroom, or is this something that's only for an after-school club? I'd love to hear your ideas! Do get in touch on Twitter **@computingchamps**. (HW)

What comes up when you search for your name?

# DIGITAL FOOTPRINTS: "I WOULDN'T WANT SOMEONE FROM SCHOOL TO SEE ALL MY POSTS"

**John Parkin** shares guidelines for curating a professional online presence

**When was the last time you searched for your own name online? This is a question I asked the trainee primary school teachers I lecture, at the start of the last academic year. Over half had not searched for their name within a year, and one in seven had never done so.**

Without an up-to-date picture of their online presence — known as their digital footprint — these students didn't know what image of themselves they were showing to the world. Nearly everyone has a trail of data online which reveals information about them, including photographs and comments made on social networking sites.

While maintaining a professional digital footprint is important for everyone — there are numerous stories of people losing their jobs, or missing out on new roles, as a result of ill-considered posts or images — it's even more important for teachers and school staff. Colleagues, students, and the parents of students may well all search for teachers' names online.

To help our trainee teachers, I researched and developed a workshop about how to create digital footprints that reflect them in a professional light — something that can often be a challenge when there is a blurring between the professional and personal online. Here are the key takeaways from the workshop.

## Search for your name

You probably haven't looked for yourself for years, but it is useful to do this regularly, so

Everyone who's online has a data trail known as a digital footprint


JOHN PARKIN

John is a senior lecturer in education and course leader for the BA Accelerated Primary Education Studies course at Anglia Ruskin University in Cambridge.


Digital resources can help you keep a professional online profile

that you can see what information is available about you when people search for you online. You can then take steps to remove any details you don't want to be shared.

## Privacy settings

It's worth spending time looking at the privacy settings you have on accounts such as Twitter and Facebook. Privacy settings can often be bewildering, but the South West Grid for Learning has produced some excellent guides to social media privacy settings (**helloworld.cc/social**). You might want to make your posts private, so that only friends and family can see your photos, videos, and comments.

## Have conversations before problems arise

It can be worthwhile having conversations with friends and family about not posting anything which could be embarrassing. It is worth having this conversation before anything goes wrong!

## Know where to get help

Social media websites should be able to help if you find content posted that is not appropriate. They will signpost how to make reports, and the Professionals Online Safety Helpline can help teachers and school staff manage a number of problems, including risks to professional reputations (**helloworld.cc/onlinesafety**).

> "LOOK FOR YOUR NAME REGULARLY TO SEE WHAT INFORMATION IS AVAILABLE WHEN OTHERS SEARCH FOR YOU ONLINE

## Change your name

When you set up social media accounts, you may not want to use your actual name, instead choosing a nickname or just a few letters from your name. This will help to stop parents and pupils searching for you online and asking to be your friend! It is still worthwhile following the other guidelines for keeping a professional online profile, even if you choose to change your name.

## Read children's books

A number of brilliant picture books have been written to help children learn about digital footprints, but they can also be used as starting points for discussing the issue with adults. *Goldilocks: A Hashtag Cautionary Tale* by Jeanne Willis is an excellent book, in which Goldilocks takes photos of her inappropriate behaviour which she shares on social media (**helloworld.cc/goldilocks**).

## The benefits of being online

It is important to remember that it is not all doom and gloom! You can manage a digital footprint so that it presents a positive and professional image and showcases your skills and talents. You can use social media sites to set up a professional learning network (PLN), which can be an excellent way to connect with teachers around the country, and even the world, to find out what other educators are doing in the classroom. I have picked up many useful tips and ideas for classroom teaching from both Twitter and Instagram. If you want to develop a PLN, you might want to set up a specific professional account so that there is no blurring of lines between your personal and professional identities.

## What did the students say?

The students who took part in the digital footprint workshop reported that they found it very useful. One student said: "When I looked at my accounts, and I could see everything on mine, I wouldn't want someone from the school I volunteer at to then do that and see all my posts. So I've made my Facebook more private, and now they can't actually see much."

I hope you enjoy using these tips, and that they help you to ensure you leave a digital footprint you can be proud of. (HW)

# UNVEILING MISCONCEPTIONS IN NUMBER BASES

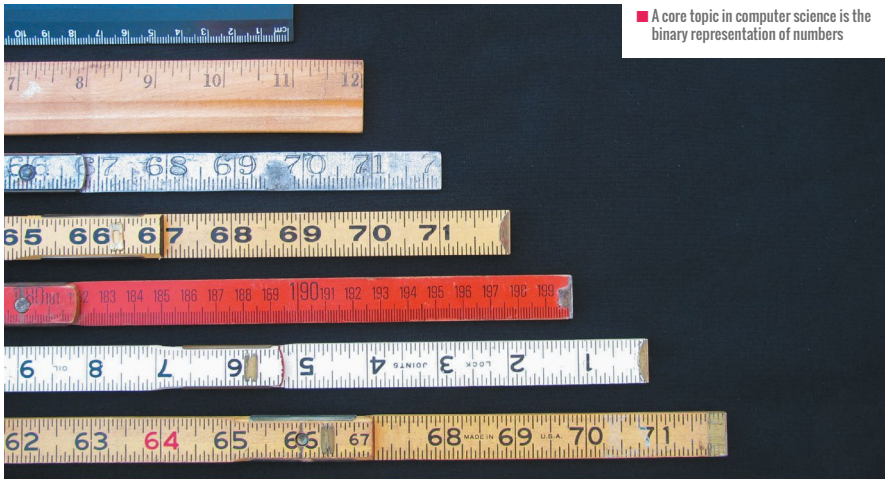
**Diane Dowling** shares how students' incorrect attempts at questions are enabling a deeper understanding of the best ways to teach A level computer science

**As students across the world are seeing rolling school closures, platforms such as Isaac Computer Science (Isaac CS), developed to support A level students, have become invaluable tools to support students' learning. But as well as providing learning materials, Isaac CS uses a strong research approach to refine its delivery of online learning. A key part of this is using students' answers to questions to develop effective feedback.**

A core topic in computer science is the binary representation of numbers. Students will be familiar with this from their previous studies, and at A level their knowledge and understanding is developed further.

As well as writing content to cover all of the learning objectives, we create quick questions that are embedded within the learning material, allowing the student to make a quick check on their learning. When the student has attempted the quick question, they can reveal the answer and the worked solution.

Unlike quick questions, end-of-topic questions do not provide a mechanism for showing the answer — instead, feedback is provided for wrong answers. Users are required to attempt the question; if the answer is wrong, they are offered structured feedback to eventually lead them to a correct answer. This feedback provides appropriate clues so that they can progress to the right answers in a fairly short time, and is carefully designed to help the student answer the question correctly. It will usually tackle common alternate conceptions — where students have frameworks of knowledge that are inconsistent with the widely agreed truth — so that they can be highlighted and addressed.



A core topic in computer science is the binary representation of numbers

The team working behind the scenes at Isaac CS can analyse the wrong answers to lead to a better understanding of alternate conceptions. We can also use this analysis to direct improvement of the Isaac CS content, so that these issues are addressed early in the learning process. Common wrong answers also highlight where a question is ambiguous or lacks clarity, and this means improvements can be made to the question.

## Wrong answers can be useful

Of course, we rarely manage to anticipate from the outset every wrong answer that students might give. Every month, we analyse a report of wrong answers. This shows us which wrong answers have been given and how many learners submitted each wrong answer.

For example, take the following question: "The binary number 10000101000101 is represented as a floating point number with a 10-bit mantissa and a 4-bit exponent. The mantissa and exponent are both stored using two's complement. Convert the number into denary." Following the method of evaluating the mantissa and exponent, explained on pages 68 and 69, using the place values shown in **Figure 1**, the correct answer is $-0.9609375 \times 2^5 = -30.75$.

Alternatively, we could use the method of moving the binary point, and as the exponent is 5, we would move the binary point 5 places to the right (**Figure 2**).

This gives us -32 + 1 + ¼ = -30.75.

This question, which requires students to type a numeric answer on Isaac CS, has had 1,689 attempts on the platform, of which 90 gave the wrong answer as 33.25.

Why do so many learners specify 33.25? The first thing we spot is that the correct answer is negative and the incorrect answer is positive. Here, the learner has not assigned a negative place value to the most significant bit. Having successfully decoded the exponent as +5, they have moved the binary point (correctly) five places to the right, but have then used an incorrect place value for the most significant bit (**Figure 3**). This causes them to arrive at 32 + 1 + ¼ = 33.25.

Having worked out the reason, we can add meaningful feedback for the wrong answer on the platform. This will ensure that, when this wrong answer is given, the learner will receive helpful feedback that will allow them to address the misconception and inform their learning.

**Figure 1**

| Mantissa | | | | | | | | | | | Exponent | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| -1 | . | 1/2 | 1/4 | 1/8 | 1/16 | 1/32 | 1/64 | 1/128 | 1/256 | 1/512 | -8 | 4 | 2 | 1 |
| 1 | . | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |

**Figure 2**

| -32 | 16 | 8 | 4 | 2 | 1 | . | 1/2 | 1/4 |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 1 | . | 0 | 1 |

**Figure 3**

| 32 | 16 | 8 | 4 | 2 | 1 | . | 1/2 | 1/4 |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 1 | . | 0 | 1 |

> ISAAC CS RESOURCES CAN BE ACCESSED FOR FREE FROM ANYWHERE IN THE WORLD

## More alternate conceptions

For this specific question, we have been able to identify five common mistakes that learners make. As well as the one shown above, we have seen:

- Students fail to move the binary point from the given start point. Instead, they move it from the beginning of the number (so in this instance they move it only 4 places). This results in a wrong answer of -15.375, or 16.625 if they compound this error with the incorrect place value of the most significant bit.
- Students fail to subtract the decimal part from the integer part. They correctly subtract 1 from -32, but then incorrectly add the fractional part, giving a wrong answer of -31.25.
- Students assume that the 1 at the start of the mantissa of a floating point number is treated like sign and magnitude (and makes no contribution to the value of the number), so they evaluate the number as -1.25.
- Students sometimes make careless arithmetic errors, for example concluding that -32 + 1.25 = -31.75.

In Isaac CS we have attempted to use a wide range of question types in order to better test a deep understanding and provide a variety of approaches. This ensures students are sometimes faced with alternative question types, in which they are not able to just repeatedly guess each multiple-choice answer until they get it right. One such type is free text, where there is one right answer but it may be entered in many different ways, and possibly in different orders. We are constantly working to improve such questions by updating the list of right answers, and the common wrong answers allow us to improve this list.

The Isaac CS platform is still in its infancy. The final topics to achieve full coverage of the specifications went live in April 2020. We are now in the review and optimise stage of the project, and questions take centre stage in this. Everyone who submits a wrong answer is playing their part in developing a learning tool that is available to computer science students and teachers at all corners of the globe.

All of the Isaac CS resources are available to access for free from anywhere in the world at **isaaccs.org**.

Wrong answers on Isaac CS play a part in developing a valuable learning tool

# CONVERTING BINARY TO DENARY

In denary (the base 10 system we are familiar with), we use 10 digits (0 to 9) to represent a number. Each digit has a specific place value that is a power of 10, such as ones ($10^0$), tens ($10^1$), hundreds ($10^2$), and thousands ($10^3$). Similarly, in binary, each digit has a place value that is a power of 2. The table below shows the place values for 8-bit numbers (**Figure 4**).

To convert an 8-bit binary number into denary, you can start by writing the binary digits in the place values table. Then, add together the place values of the binary digits that are set to 1.

For example, to convert the binary number $11001101_2$ into denary, you would perform the following calculation: $128 + 64 + 8 + 4 + 1 = 205_{10}$. The subscripts $_2$ and $_{10}$ denote binary and denary notation, respectively (**Figure 5**).

## Signed integers and two's complement

Whole numbers such as 7, 12, and 3988 are called integers. Unsigned integers have positive values by definition, while signed integers can be positive or negative. In denary, negative integers are represented using a minus symbol before the value of the number, for example –19. In computer systems, two's complement is one way to represent signed integers in binary.

Using two's complement, the most significant bit has a negative place value. This way, negative numbers expressed in binary start with a first bit of 1, and positive numbers start with a first bit of 0. For example, using two's complement representation gives us the place values for a number with 6 bits shown in **Figure 6**.

## Fixed point fractional binary numbers

Fractional binary numbers can be represented in fixed point or floating point form. In fixed point form, the binary point is set in a fixed position, and therefore it does not need to be stored in memory.

Consider the binary number $01111011_2$, which we are told is an unsigned fixed point form with 4 places before the binary point and 4 places after the binary point. We can write down the place values for the binary number. The place values after the binary point are halved every time. As the number is unsigned, the most significant bit has a positive place value (**Figure 7**).

By adding together the place values that are set to 1, we can express the number in denary as $4 + 2 + 1 + {}^1/_2 + {}^1/_8 + {}^1/_{16} = 7.6875_{10}$.

## Binary floating point numbers

Binary floating point numbers are expressed in the form mantissa × $2^{\text{exponent}}$, for example $0.101 \times 2^4$.

If the binary number $0.101 \times 2^4$ is stored in a format that allows 8 bits for the mantissa and 4 bits for the exponent, it would be stored as a 12-bit number (**Figure 8**).

The two's complement representation is used for both the mantissa and the exponent. The two's complement mantissa allows positive and negative values to be stored. The two's complement exponent allows both positive and negative exponents to be stored, which means that very large and very small values can be represented.

## Converting a binary floating point into a denary

One way to convert a floating point binary number into denary is to use the expression mantissa × $2^{\text{exponent}}$ and evaluate each part of it, taking into consideration that both the mantissa and the exponent are represented in two's complement.

For example, the floating point number $101100000010_2$ is represented using 8 bits for the

**Figure 4**

| $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
|---|---|---|---|---|---|---|---|
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

**Figure 5**

| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |

**Figure 6**

| -32 | 16 | 8 | 4 | 2 | 1 |
|---|---|---|---|---|---|

**Figure 7**

| 8 | 4 | 2 | 1 | 1/2 | 1/4 | 1/8 | 1/16 |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

**Figure 8**

| Mantissa | | | | | | | | | | Exponent | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| -1 | . | 1/2 | 1/4 | 1/8 | 1/16 | 1/32 | 1/64 | 1/128 | | -8 | 4 | 2 | 1 |
| 0 | . | 1 | 0 | 1 | 0 | 0 | 0 | 0 | | 0 | 1 | 0 | 0 |

**Figure 9**

| Mantissa | | | | | | | | | | Exponent | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| -1 | . | 1/2 | 1/4 | 1/8 | 1/16 | 1/32 | 1/64 | 1/128 | | -8 | 4 | 2 | 1 |
| 1 | . | 0 | 1 | 1 | 0 | 0 | 0 | 0 | | 0 | 0 | 1 | 0 |

Isaac CS offers structured feedback that will eventually lead students to the right answer

mantissa and 4 bits for the exponent (**Figure 9**).

Using the place values, we can see that the denary value of the exponent is 2, and the denary value of the mantissa is $-1 + {}^{1}/_{4} + {}^{1}/_{8} = -0.625$. The value of the number in denary is then $-0.625 \times 2^{2} = -2.5_{10}$.

An alternative method to convert a floating point number into denary is to use the value of the exponent to move the binary point. The mantissa represents a value with a binary point between the two most significant bits. The exponent determines how far to 'float' the point to determine the final value of the number.

If the exponent is positive, the final value will be larger than the mantissa, so the point will 'float' to the right and the number of places it will move is determined by the value of the exponent. If the exponent is negative, the point will 'float' to the left the same number of places as the exponent.

For example, if we are asked to convert $010001000101_{2}$ to denary, where the mantissa is 8 bits and the exponent 4 bits, we can start by writing down the place values for the mantissa and the exponent in two's complement, where the most significant bit determines the sign of the number (**Figure 10**).

We first calculate the exponent, which is $4 + 1 = 5_{10}$. Here, the exponent is positive, so move the binary point 5 places to the right. Then we can calculate the number using the place values table, in the same way as for fixed point binary (**Figure 11**).

This gives us $16 + 1 = 17_{10}$. (HW)

> "WE RARELY MANAGE TO ANTICIPATE FROM THE OUTSET EVERY WRONG ANSWER THAT STUDENTS MIGHT GIVE

**Figure 10**

| Mantissa | | | | | | | | | | Exponent | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| -1 | . | 1/2 | 1/4 | 1/8 | 1/16 | 1/32 | 1/64 | 1/128 | | -8 | 4 | 2 | 1 |
| 0 | . | 1 | 0 | 0 | 0 | 1 | 0 | 0 | | 0 | 1 | 0 | 1 |

**Figure 11**

| -32 | 16 | 8 | 4 | 2 | 1 | . | 1/2 | 1/4 |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 1 | . | 0 | 0 |


## DIANE DOWLING

Diane is a learning manager at the Raspberry Pi Foundation, where she works on the Isaac Computer Science platform. In her spare time, she is a trustee of a national charity that runs robotics events for sixth-formers.

# CREATE A RELAXING INTERACTIVE SCENE

**AGE RANGE**
7-11 years

**YEAR GROUP**
Years 4-6

**LESSON TYPE**
Block-based coding

**REQUIREMENTS**
- A computer or tablet capable of running Scratch
- Scratch 3 (either online or offline)

Serene Scene is part of a new Scratch project pathway from the Raspberry Pi Foundation that encourages learners to take care of their well-being

**The Raspberry Pi Foundation's new pathway of digital making projects focuses on well-being. There are six projects in the pathway, which include themes from meditation and relaxation to fitness. The pathway is available at helloworld.cc/look-after-yourself.**

Young people need to learn techniques for coping with life's everyday challenges so that they can maintain healthy minds. Rather than using technology as a communication tool, the new pathway encourages learners to harness technology to connect with themselves. By working through the projects, they can develop a sense of calm, as well as gaining a feeling of accomplishment.

In Serene Scene, young people develop a relaxing interactive animation. The project includes a number of variable sliders that allow the user to alter the appearance and behaviour of flora and fauna in the wood. They can also modify the ambient sounds. Through this project, learners develop their knowledge of key computing concepts, such as repeat forever loops, creating and changing the value of variables, and using operators. You can preview the completed project at **helloworld.cc/serene-scene-complete.** (HW)

## OBJECTIVES

- ✔ How to create variables
- ✔ How to use sliders to change the value of variables
- ✔ How to use forever loops to check variable values

## RELEVANT LINKS

**helloworld.cc/serene-scene**
**projects.raspberrypi.org**

## ACTIVITY 1: GROW A TREE 15 MINUTES

If you are working online, open the starter project in Scratch (**helloworld.cc/serene-online**). If you are working offline, download the project starter file (**helloworld.cc/serene-offline**).

You should see a forest scene with a tree, flowers, and a grasshopper. First, you will change the size of the tree. Select the **Tree1** sprite in the Sprite list below the Stage.

A variable is a way of storing numbers and/or text. To create a new variable in Scratch, click on the **Variables** menu to the left of your screen, then click the **Make a Variable** button. You can give your variable a name. Call this variable 'tree'. You should now see five new blocks that you can use — see Figure 1.

You will also see that the 'tree' variable is visible on the Stage. There are many ways to control the value of a variable, but in this project, you will use sliders.

On the Stage, right-click the 'tree' variable, and a drop-down menu will appear. Select **slider** from the menu. Move the slider forwards and backwards, and you will see that the value of the 'tree' variable changes between 0 and 100 (percent). Now, you will use the value of the 'tree' variable to change the size of the tree.

First, use a **when green flag clicked** block with a **forever** loop. Add a **set size to** block into the loop. This means that once the flag is clicked, the **set size to** block in the **forever** loop will keep the tree size at 100 percent — see Figure 2.

Now, add the 'tree' variable into the **set size to** block, as shown in Figure 3. You can now move the slider to adjust the size of the tree.

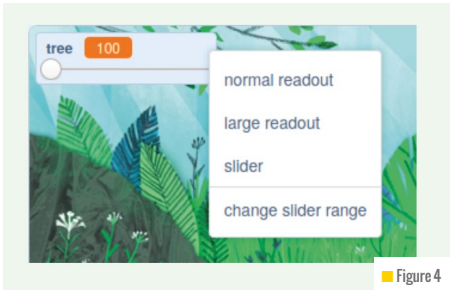


Figure 4

At the moment, the tree size can only be changed from 0 to 100. On the Stage, right-click on the 'tree' slider and select **change slider range** from the drop-down menu — see Figure 4. Change the range to between 100 and 300. Now, move the slider to watch your tree grow in size from 100 percent to 300 percent.

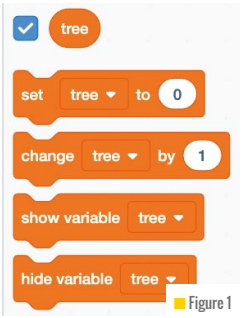


Figure 1

Figure 2

Figure 3

## ACTIVITY 2: COLOUR THE FLOWERS 15-20 MINUTES

You used a 'tree' variable to change the **size** of the tree. Now you can use a variable to change the colour of the flowers. Click on the **flowers** sprite, then create a new variable called 'flowers'. Go back to the previous step for a reminder if you need to.

Change the **color effect** of the flowers, in the same way that you changed the size of the tree. Then add in your 'flowers' variable — see Figure 5.

Variables that store numbers do not always have to have values that are greater than 0. You can use negative numbers as well. On the Stage, right-click on the 'flowers' variable and set it to slider. Now change the slider range to between **-100** and **100**. Click on the green flag, then adjust your flowers slider to see the flowers change colour.

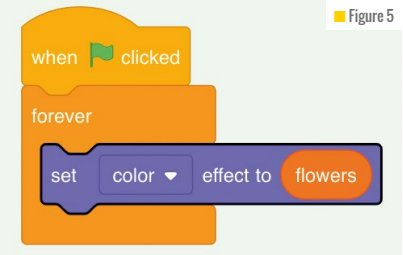


Figure 5


### EMMA POSEY

Emma is a learning manager on the Informal Learning team at the Raspberry Pi Foundation. She also develops content for the National Centre for Computing Education's research project, Gender Balance in Computing (**@emmaposey**).



### MARC SCOTT

Marc was a teacher of science, computer science, and systems and control for 15 years, before joining the Raspberry Pi Foundation, where he now works as a senior learning manager.

## ACTIVITY 3: SET THE SCENE 15-20 MINUTES

In Scratch, you can also add code to the **Stage**, rather than to a sprite. In this step, you will change the brightness of the backdrop and add some soothing sound effects.

Go to the Stage pane and click on the **backdrop** — see Figure 6. You will now develop a program for the **backdrop** in the Code area. In the **Variables** blocks menu, click on **Make a Variable** and call the new variable **light**. Change the variable to a **slider** with a range between **-40** and **40**.

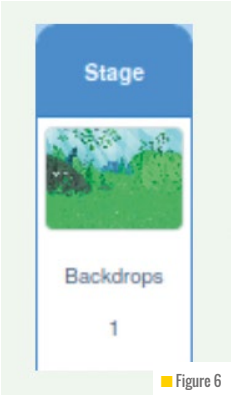


Figure 6

Figure 7

Just as before, you can use this variable to change the appearance of the backdrop. Use the **set color effect to block**, but use the drop-down menu to change **color** to **brightness**. See Figures 7 and 8.

Now, when you adjust the **light** slider, you should see the forest change its brightness.

You can also add sound effects to the **Stage**. To do this, click on the **Sounds** tab at the top left of your screen. Click on the **Choose a Sound** icon in the bottom left-hand corner of the screen to select a sound. You can now search for sounds. In this project, we will use the **Rain** sound. Click on the **Rain** icon to select the sound. Create a **rain** variable and make it appear as a slider. Add code to **play** the **Rain** sound **forever**, and **set** the **volume** of the sound to the value of the **rain** variable. See Figure 9.

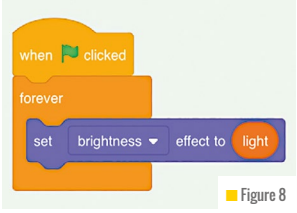


Figure 8

Figure 9

Click on the green flag to run your code and change the volume of the rain with the slider.

## ACTIVITY 4: ADD SOME FAUNA 15-20 MINUTES

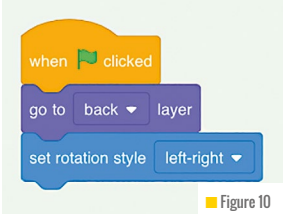


Figure 10

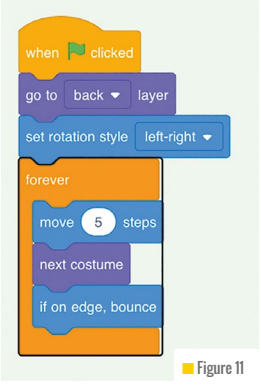


Figure 11

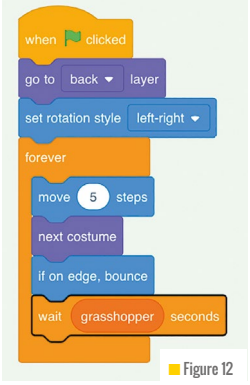


Figure 12

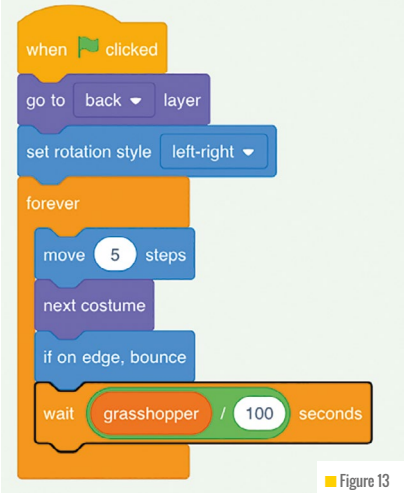


Figure 13

In this step, you will add a moving grasshopper to your serene scene. Set up the **Grasshopper** sprite so that it moves from left to right and appears behind the tree and flowers. See Figure 10.

Now, make your **Grasshopper** sprite move back and forth across the Stage, as shown in Figure 11.

The **Grasshopper** sprite is moving a little quickly at the moment, but you can use a **variable** and a **wait** block to slow it down. Create a new **variable** called **grasshopper** and switch it to a **slider**. Now, you can use a **wait** block to slow the grasshopper down, as shown in Figure 12.

If you click on the green flag and move the slider, you will notice that the **grasshopper** moves very slowly. To fix this, you can multiply the grasshopper variable by a number smaller than 1 (in other words, a number that is smaller than 100/100). Go to the **Operators** blocks menu and find the **/** block. Now, drag this into your script to divide the **grasshopper** variable by **100**. See Figure 13.

When you adjust the slider, the grasshopper will move at a different speed. You might also like to adjust the **slider range** down to between **0** and **20**.

# FILMS IN COMPUTING CLASSES

**Alan O'Donohoe** argues that watching a movie can be more than an end-of-term activity, and shares suggestions for how to incorporate films into lessons

**Join me in a little game for a moment. I want you to imagine you are planning a trip to the cinema for yourself and a group of your students. You have been asked to select a film for screening related to technology, computing, or computer science. What film would you choose?**

If you were allowed only seconds to respond, you might come up with one. When I ask others this question, the most popular answers have been *Hidden Figures*, *The Imitation Game*, and *WarGames* — and I explain the various merits of these suggestions further on. I suspect, however, that the more time and thought you give this question, the more difficult the selection becomes, so that narrowing it down to one single title becomes increasingly difficult.

There used to be a culture in schools, where at the end of term or in exceptional circumstances, the teacher in charge would stick on a DVD. I remember, when covering some classes for a colleague who taught history, I saw that colossal beach scene from *Saving Private Ryan* five times in the same day! One school I taught in instructed teachers to refrain from showing films at the end of term because they felt "teachers were abusing it", and that "pupils preferred to be taught right up until the last lesson of the term, in spite of their protests".

Yes, of course, there is the potential for exhausted teachers to fall back on a DVD at the end of term or in emergencies, but that doesn't mean we should give up on ever watching any films in lessons. We should be encouraging our students to be reflective reviewers, not just passive consumers of media, and creating opportunities for them to critically evaluate the portrayal of computing and technology in films and on TV. Our students will be exposed to many more hours of film, drama, and documentaries than we teach them for. So there's a real opportunity for us to equip them with the skills and expertise not only to enjoy a wide selection of film and documentary titles, but to know which ones are worth watching, and why.

Occasionally I hear other computing teachers complain that, compared to other school subjects, there really aren't many films, dramas, and documentaries that are relevant to computing education and computer science. **While true** (CS joke!), those that are available are very accessible. Your history teaching colleagues will probably be the last to complain about the limited number of film titles to choose from — but we can afford them some sympathy, as they can't get anywhere near as enthusiastic about exciting new developments in their subject area. Yes, there is a comparatively narrow choice of computing-related titles to choose from. However, we do have some fantastic films that we can use to add more excitement, depth, and interest to our teaching.

In this guide, I plan to highlight the opportunities that film, drama, and documentaries offer us as a teaching resource. I'll also suggest practical approaches that you might readily use to develop critical media literacy in your students, while fostering an interest in computing technology development.


Films can add more excitement and depth to computing lessons

## How can film help teachers?

Of course, it's prudent to start by asking ourselves, "Why are we doing this?" in relation to any activity we embark upon. However, I would strongly caution you to prepare your argument more strongly in the case of showing a film. This is particularly important if you sense any risk that your plans to use film to enrich the curriculum may be misinterpreted or misrepresented by colleagues, parents, or students as an attempt to look for an easy activity to plan.

To help strengthen your case, here are some compelling arguments to convince you and others:

- **Engaging and inspiring learners:** While writing this guide, we're in yet another lockdown and teachers have been reporting that they are burning out, while their students are losing interest in 'stale' lessons. I tend to go on at length about the need for enrichment in our subject area, with real justification. Without enjoyable and interesting activities, students and teachers can perceive our subject as being very dry. Reviewing a film can add a very different dimension to learning.
- **Increasing exposure to computing:** In many schools, computing is timetabled for 45–60 minutes each week up to the age of 14. Post-14, that may increase to two or three hours a week for those who choose it as an option. Suggesting that students watch a two-hour film at home one week substantially increases that amount of time.
- **Discussing difficult matters:** Films like *Hidden Figures* and *The Imitation Game* help to contextualise topics like prejudice, diversity, and inclusion, and provide examples that students may relate to more easily during a discussion in lesson.
- **Application of computing:** Developments within computing have led to huge advances in fields such as CGI, scripting, editing, VFX, and SFX. Computing has increased production capabilities and reduced budgets.
- **Accessibility:** Unfortunately, some of our favourite learning resources in computing can be inaccessible to students due to software licensing restrictions, hardware availability, or connectivity. On the other hand, many classrooms and households can access film through the medium of broadcast TV, video-sharing platforms, streaming media, and on-demand services. Many recommended computing films can be purchased on DVD for £1–3 from online auction sites, discount stores, and charity shops.

> "WE SHOULD BE ENCOURAGING STUDENTS TO BE REFLECTIVE REVIEWERS, NOT JUST PASSIVE CONSUMERS OF MEDIA

- **Rewarding students:** At certain times of the year, it's helpful to reward groups of students with a trip or a treat as recognition; a carefully selected film such as *WarGames* or *Ralph Breaks the Internet* could provide such a reward when a trip out of the classroom is not practical.
- **Plan B:** Computing teachers know only too well how important it is to have a contingency plan in place for those times when there is no teacher, no heating, no computer, no network, no timetable, or most of the class missing. Having a couple of DVDs to hand can be a valuable lifeline, providing they're not used too often, and you remembered to put the disc back in the case!


Many themes explored in *WarGames* still hold true today

## Spotlight on three films

Here are three films that every computing teacher should have watched at some point or other. If you haven't seen all three of these, set yourself some homework to plan how and when you will make the time for them:

- ***Hidden Figures* (PG, 2017)** This film has broad appeal, making it suitable for non-computing audiences too. It lends itself very well to other themes, like space and STEM, but particularly to diversity and inclusion themes, such as Black History Month, Black Lives Matter, and Ada Lovelace Day. Clips from the film would be suitable for use in an assembly. The film is based on the excellent book by Margot Lee Shetterly and presents inspiring stories of four Black women 'human computers' who had to counter prejudice in their work at NASA, and celebrates their achievements. Like any film, some scenes and dialogue are fictionalised — but the film does not stray too far from facts and actual events. The film is rated PG, and I have used extracts of the film with children as young as ten years old.
- ***The Imitation Game* (12A, 2014)** Regrettably, this film probably receives far more praise and viewings than it ought to — to the extent that some codebreaking experts jokingly refer to it as *The Irritation Game*. Without doubt, Benedict Cumberbatch's portrayal of Alan Turing is beyond brilliant, but aspects of the Bletchley Park story deviate from historical fact. I think we just have to learn to love this film, accepting the fact that many of the children we teach will be exposed to the film at some point. Our best strategy is to embrace that as an opportunity and ensure that our students are well informed enough to appreciate the film critically. It's also a useful entry point to introduce a potential visit to Bletchley Park and/or The National Museum of Computing.


#exaflicks is an online film club for computing enthusiasts

- ***WarGames* (PG, 1983)** This film is an incredibly popular title among teachers and enthusiasts of a certain age — many have revealed the extent to which the film helped inspire them into their chosen career. I'll admit that I, myself, wasn't convinced it would be as warmly received by students — but I have been proved wrong. The film has dated exceptionally well and has a certain 1980s nostalgic feel to it. There are many themes in the film that still hold true today, and it would serve well as a light reward reserved for end-of-term lessons. Some computing classrooms I have visited feature *WarGames* posters as well as quotes and scenes from the film.

Carefully selected films can be used as rewards

## A case study: Hidden Figures

Working with computing teacher Nic Highes, we planned to show the *Hidden Figures* film to Year 6 (ages 10–11) classes over two 45-minute lessons for each class. We decided, rather than to simply play the film from start to finish, we would instead play selected clips from the film to facilitate discussion in class. We designed activities for the children to raise the issues around prejudice, diversity, and inclusion.

In the first lesson, we shared some still photos from historic space missions and asked children to name as many astronauts or space scientists as they could. As we predicted, the names children provided were almost entirely all white men — this provided a good opportunity to introduce the film title *Hidden Figures*, which tells the tale of a group of Black female scientists who are less well known. Then we explained that we were about to watch a scene from the film at the Langley Research Center, but we wanted the children first to imagine they were working there. We asked them to describe what they might see around them, what noises and voices they might hear, and how people might be dressed. The scene is the one where Al Harrison asks his assistant, "Ruth, what's the status on that computer?" to which Ruth responds, "She's right behind you, Mr Harrison." He turns around to see Katherine Johnson for the first time – the new 'computer' – and his face shows a mixture of emotions. We asked the children what his thoughts might be, and why. This then seemed an opportune moment to link to the civil rights movement in the US, as well as themes of segregation and prejudice.

In each of the two lessons, the activity followed similar patterns: we asked students to make predictions, explain the deeper meanings of scenes, explore why this was so, and look at how things were changing for the better. We also read some abstracts from the Margot Lee Shetterly picture book of the same name. We set the children an assignment and asked them to design motivational postcards that might summarise the film's message, celebrate the scientists featured in the story, or inspire others to counter prejudice.

## THINGS TO CONSIDER

- **Accessibility:** Many of the popular titles have trailers and clips available online that lend themselves more easily to use in class.
- **Copyright:** As teachers, we're constantly reminded that it's not enough for us to simply observe legislation: we must be actively seen to uphold the law. This is where, in the UK, an ERA licence can help. Excepting independent schools and 16-19 schools and colleges, local authority and maintained schools will almost certainly have an ERA licence in place, but don't just assume so - a colleague will be able to confirm this for you. In my last school, the certificates were on display in the admin offices and annually refreshed. More guidance via the Department for Education here: **helloworld.cc/copy**.

© Photo by Alex Litvin on Unsplash

> "WE ASKED PUPILS TO DESIGN POSTCARDS THAT CELEBRATE THE FILM'S SCIENTISTS, OR HELP TO COUNTER PREJUDICE

## #exaflicks – a monthly film club

Have you considered hosting a regular film club or festival for your students, colleagues, or friends? You might find the whole idea daunting right now, but why not start with a popular title first and

take it from there? You could be pleasantly surprised at how positive the response is from everyone.

When we experienced the first nationwide lockdown in March 2020, I was looking for positive ways to help keep teachers talking to each other, to help support teacher well-being, but also as a distraction from the news and the discussions about heavy workload. I decided to try hosting a monthly watch party online, which we called #exaflicks. We started with some of the obvious titles, like *The Imitation Game* and *Hidden Figures*, but we've also included some less obvious choices, like *Ring of Spies*.

Each month I invite a panel of five special guests to join us as we first watch, then discuss the film. Due to copyright restrictions, it's more typical for guests and audience participants to watch the selected title in advance, but we meet online to discuss the title together. You can learn how well this works by watching recordings from these panel discussions on the exa.foundation YouTube channel (**exa.is/channel**) — the recordings are really intended for teachers or other adults, but you might choose to play part of one to a class, and perhaps ask them why they think a particular guest made a certain comment, or whether the class would agree with that comment.

Do you have a computing-related film to recommend? Are you looking for more ideas of activities you might use to enrich the curriculum? Would you like to join a future #exaflicks panel discussion? If so, please contact Alan O'Donohoe at exa.foundation for more information: alan@exa.foundation. (HW)

## TEACHERS' FAVOURITE COMPUTING FILMS

Beverly Clarke, CAS Community Manager, explains the value she finds in film as a teaching resource:

"During my teaching days, I found that some pupils were disengaged with computing lessons, so I decided to look for ways to engage them that they were already connecting with. The after-school cinema and film club I set up was one such way.

"I would recommend *Coding* from the *Explained* docuseries on Netflix. The film is narrated by supermodel Karlie Kloss, who many pupils may be aware of, so immediately there is a relatable figure in the real world with an interest in this topic. It starts off with an example of how a coding mistake affected an emergency call system in real life. This then opens debate on ethical responsibilities. Within the same series, logic gates and their function are discussed, and computational thinking and algorithms are explained too. The film also provides lots of points to discuss the importance of testing products and the moral and ethical implications of poorly tested products. It takes us right up to the present day with machine learning and image recognition."

Martin Sexton, a computing teacher, told me about his favourite computing film, *The Social Network*: "It offers a lot of potential for use with students. There's lots of close links to the curriculum, particularly ethics and legislation."

Dean Wild, a computing teacher, said, "Recently I watched the docuseries on Disney+ of the making of *Frozen II*. It was fascinating watching how all the departments work together during production, from writers and animators, through to effects and lighting. Well worth a watch - I've recommended my students to watch it if they have a Disney+ subscription."

Computing teacher Theresa Russell recommended her students watched *WarGames* when it was broadcast on TV recently: "Lots of my students watched it with their parents too - it was great. One of my GCSE students, Jessica, wrote a really good review exploring the laws and ethics of how it happened."


### ALAN O'DONOHOE

Alan has over 20 years' experience teaching and leading technology, ICT, and computing in schools in England. He runs exa.foundation, delivering professional development to engage digital makers, supporting computing teaching, and promoting the appropriate use of technology (**@teknoteacher**).

# SURVEY RESULTS

Since the last issue of Hello World was released, we continued to send out a short monthly survey to random samples of Hello World subscribers.

At the end of last year, we also carried out some more in-depth interviews with readers to find out what they find useful in the magazine — and how we could improve.

Here, we are sharing some of the comments we have received. We read every comment — so if you get an email asking for your views on the magazine, please do take five minutes to let us know what you think. Or if you'd like to get in touch at any time, email contact@helloworld.cc. We are always interested in reader feedback.

And if you would like to write for the magazine, please do drop us an email too!

*Our surveys are managed and analysed by Joshua Crossman.*

I really enjoy the magazine, the physical mag gets passed around, not sure PDF does the same job, so keep printing!

**The best part of Hello World for me is that there are realistic ideas for lessons on different areas of computing, that I can actually use in my lessons. Definitely more of this would be appreciated in other topics not discussed much so far, such as networks, cryptography/encryption, and boolean logic.**

The amount of CPD I get from the magazine is really valuable. The work on computing and SEN came through at a really timely juncture for me. Please keep doing what you're doing. You rock.

**I'd like to see you dedicate an issue to initiatives currently being carried out in Africa and Latin America. I feel that your audience could learn a great deal about the lengths that some educators go through in order to bring CS education to their pupils.**

Could the magazine go on tour? It'd be great to see you in Montpellier, France!

**I love the schools sharing their practice and their impact, because it can really trigger thought and innovation, an "If they can do it, why can't we do it?" type thing. I love seeing how projects can be brought into the classroom. Brilliant, great resource.**

Thank you so much for what you have done so far (the AI edition was so useful!).

**I teach in FE in Scotland and we've made a move to online by default this autumn, the resources in Hello World have been a great help with the transition out of the classroom!**

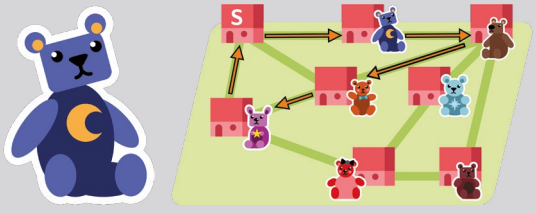
A fantastic magazine – I keep them all and frequently check them. I love it!

**I'm not a teacher anymore, but still use the resources in more informal learning environments.**

As computing teachers, we support other teachers helping them to use technology in their lessons, so more examples of how computing can help in other subjects would be great.

## BEBRAS PUZZLE SOLUTION (PAGE 81)

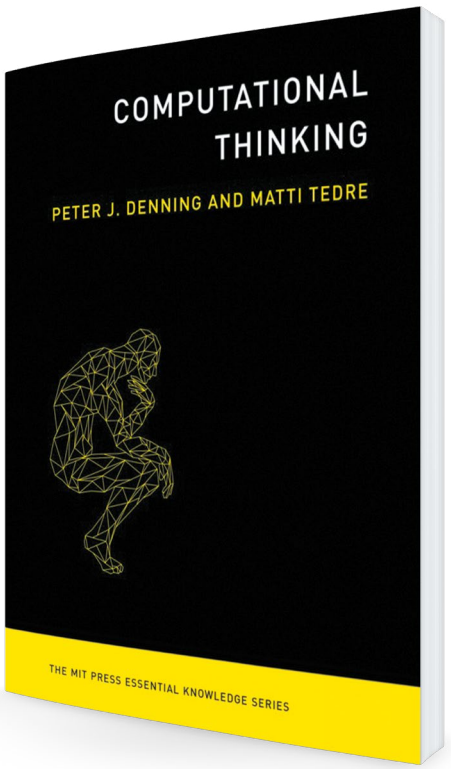
In the image, you can see one of only two routes the beaver family could have taken. The other route is the one that goes in the reverse direction. Both of these routes pass exactly four teddy bears. They therefore must have forgotten to take a photo of the purple bear with the moon on its belly.



# COMPUTATIONAL THINKING

INFO **BY** Peter J. Denning and Matti Tedre | **PUBLISHER** The MIT Press | **PRICE** £12.99 | **ISBN** 9780262536561 | **URL** helloworld.cc/comp-thinking

**Sue Sentance** tells us about a book that offers a rich and comprehensive overview of computational thinking and explores a way of thinking that predates electronic computers




**Presentationally, this is a delightful little book — accessible to read and almost pocket-sized. The title is *Computational Thinking*, but at times you would be forgiven for thinking you were reading a history of computing, as it tracks back through the history of computing to comprehensively define and describe the many complexities of computational thinking (CT):**

*"We see CT as a mental discipline for thinking about designing computations of all kinds, a skill at the advanced levels honed and improved through extensive practice and experience."* (page 191)

The authors separate out computational thinking for professionals and computational thinking for beginners as two ways of thinking about the subject, and focus primarily on the professional end of the spectrum, with sections on computational methods, machines, software engineering, computer science, design, and computational science — all revealing different aspects of computational thinking. These chapters give an excellent background to the development of our discipline.

When the authors turn to computational thinking for all, that is, teachers and students, and address the surge of interest generated by Jeanette Wing's short article in 2006, they are keen to bring up these points: that computational thinking is not transferable, based on a lack of evidence of skill transfer from computer programming to other subjects; is not a set of general skills that can be learnt through performing everyday procedures; and is different for different domains. In addition, computational thinking is *not* new (i.e. since 2006), it's not thinking like a computer scientist, and it's *not* restricted to school education (called K-12). Instead, it's very broad in scope, and predates the existence of electronic computers.

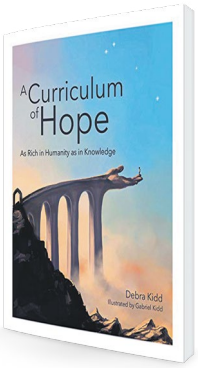
I found this book refreshing. We may have previously oversimplified computational thinking to a short tick list of skills. The result was, in my opinion, that this simplification just didn't work as a topic you could teach and assess, and now we've all gone a bit quiet about computational thinking. Do read this book, and introduce your students to the "richness, breadth, and depth of computational thinking" present in all aspects of the teaching of computing.

I recommend this book to all computing teachers and A level computer science students, too: and suggest having one in the computing section, and another in careers. (HW)

# A CURRICULUM OF HOPE: AS RICH IN HUMANITY AS IN KNOWLEDGE

INFO BY Debra Kidd | PUBLISHER Independent Thinking Press | PRICE £18.99
ISBN 978-1781-35342-4 | URL helloworld.cc/kidd

Ben Garside




**In *A Curriculum of Hope*, Debra Kidd argues that each student's education should be relevant to their experiences. She gives ideas on how to creatively deliver and meet the needs of a curriculum, without feeling limited by exam specifications or other constraints.**

The book is filled with examples of interesting classroom practice and thought-provoking questions — for example, if someone is a teacher in an international school, whose history do they teach? Kidd gives just a few examples relating to computing, but in reading the book, I was prompted to reflect on my own experiences of teaching the subject. For example, how should teaching programming or the ethics of artificial intelligence differ between a classroom in Manchester and a classroom in Mumbai?

In practice, many teachers may feel that they don't have a strong influence over what is taught in their subject area. Even fewer have influence over their school policy on the curriculum. Whatever a teacher's individual influence is in a setting, *A Curriculum of Hope* will give them practical ideas on how to provide the most meaningful learning experiences for their students. (HW)

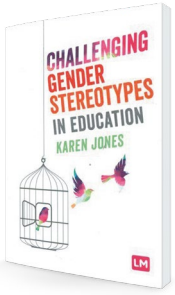
# CHALLENGING GENDER STEREOTYPES IN EDUCATION

INFO BY Karen Jones | PUBLISHER SAGE Publishing | PRICE £22.99 | ISBN 9781526494535
URL helloworld.cc/jones

Katharine Childs

**Teachers are invited in this book to reflect on curriculum content and pedagogy through the lens of gender in education, and to recognise where gender stereotypes still exist.**

The chapter *Women in computing* explores why girls are under-represented in computer science careers. Four activities outline meaningful, lasting challenges to help us reflect on the barriers to inclusivity in the computing classroom, and go beyond box-ticking checklists or short-term initiatives.




However, computing is just one subject in the curriculum, and this text examines gender stereotyping across the whole education system. The section on role play in the early years is particularly interesting when we consider how young children are when they begin to form gender stereotypes. The chapter on a whole-school approach to gender equity contains practical tips and approaches based on existing research.

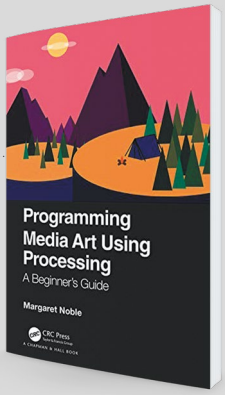
This isn't a book to read and put away. It invites thought, action, and ultimately systemic change. (HW)

## ESSENTIAL READING

**New titles that explore creativity and innovative classroom practice**

### PROGRAMMING MEDIA ART USING PROCESSING: A BEGINNER'S GUIDE

BY Margaret Noble
PUBLISHER CRC Press
PRICE £35.99
ISBN 9780367508289
URL helloworld.cc/processing




**Using the open source and creative language Processing, Margaret Noble explores visual design through computer programming and provides a series of lessons with step-by-step examples for creating media art projects.**

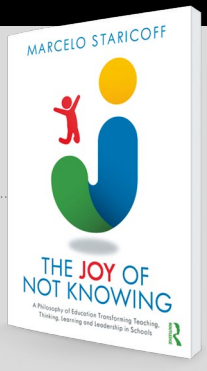
### TEACHERS VS TECH?: THE CASE FOR AN ED TECH REVOLUTION


BY Daisy Christodoulou
PUBLISHER Oxford University Press
PRICE £17.99
ISBN 978-1382004121
URL helloworld.cc/christodoulou

**Educational commentator Daisy Christodoulou examines the ed tech debate, exploring a broad range of topics and examples, and outlining how technology can ultimately be used to improve educational outcomes.**

### THE JOY OF NOT KNOWING: A PHILOSOPHY OF EDUCATION TRANSFORMING TEACHING, THINKING, LEARNING, AND LEADERSHIP IN SCHOOLS

BY Marcelo Staricoff
PUBLISHER Routledge
PRICE £15.99
ISBN 9780367172725
URL helloworld.cc/staricoff




**An innovative, theoretical, and practical guide for educators, this book outlines how all students can develop a love of learning and grow into global citizens.**

# THE BEBRAS PUZZLE PAGE

Each issue, Chris Roffey shares a computational thinking challenge for your students

## THE CHALLENGE: TEDDY BEAR HUNT

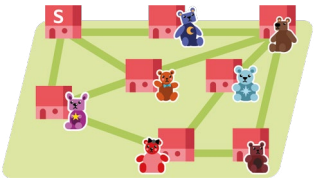
### Finding a route

**The beaver family went on a teddy bear hunt around their village, shown on the right. They started from their home, marked S on the map.**

They walked along the roads and returned to their home. Along the way, they took photos of the teddy bears they saw. They saw four teddy bears but arrived home with only the three photos below. Which teddy bear did they forget to take a photo of? *Solution on page 78.*

#### Further information

Finding a route is a common computer science problem. To solve this task, follow the routes and find out which route meets the conditions. Although this task is simple, it becomes harder to solve as the map expands.




The map is a representation of a planar graph made of nodes (houses), and edges (roads between houses). The task is to determine the node that must be on any route consisting of four nodes that are not 'home'. It is a combinatorial problem on a graph, and is like drawing the route from home to school, then to the library, to the shop, and finally back home.

**DOMAIN**

Data, data structures, and representations

**SKILLS**

Abstraction, evaluation

**AGE**

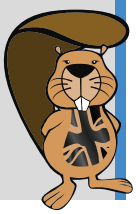
6-10 years

**DIFFICULTY RATING**

Ages 6-8 medium; ages 8-10 easy

**ABOUT BEBRAS**

Bebras is organised in over 50 countries and aims to get students excited about computing and computational thinking. Over half a million UK students took part in the last two annual Bebras challenges. Our archived questions let you create your own auto-marking quizzes at any time during the year. To find out more and register your school, head to **bebras.uk**.



## COMPUTING KEYWORD SPOTLIGHT: GRAPH THEORY

### Defining everyday techniques used by problem-solvers

**In computer science and mathematics, a group of objects and the relationships between them can be stored in a graph. These are not like line graphs or bar charts, but much more like what we might usually describe as a map.**

The objects, such as the houses in the teddy bear hunt problem above, are called the vertices, or nodes, of the graph. The relationships, for example the roads between the houses in the teddy bear hunt problem, are called edges.

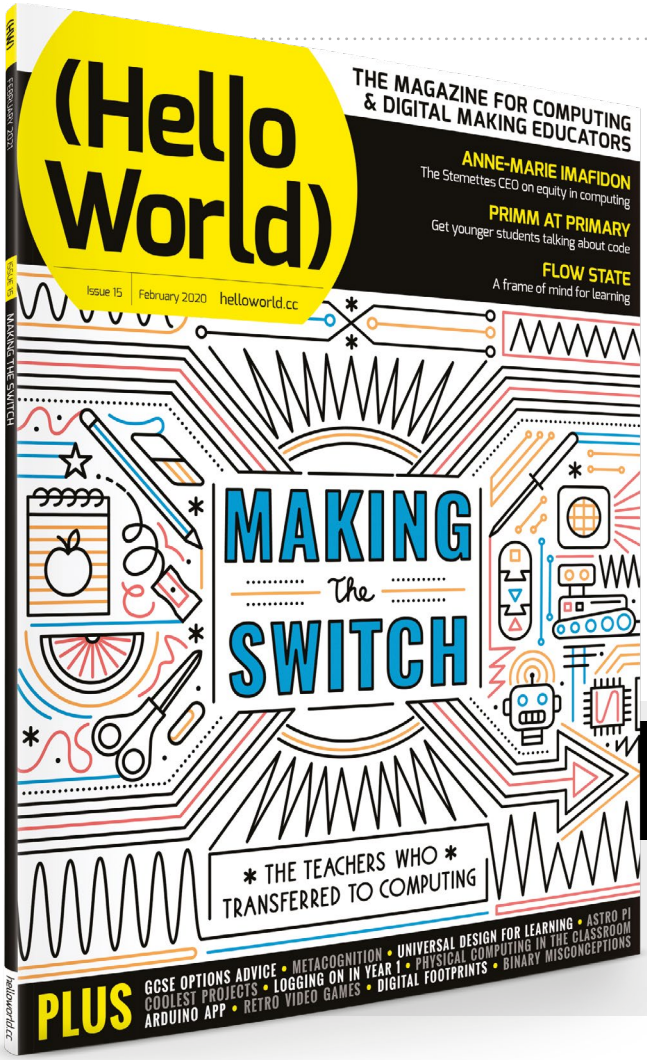
A lot of problems presented in Bebras are simplifications of different types of graphs. This is because graph theory can be used to model a huge number of situations we find in life. Consequently, computer scientists and mathematicians have studied them extensively and come up with many algorithms for solving problems that can be described with graphs, such as finding the shortest route, a route that navigates every node, and so on. An example of a famous problem that involves navigating graphs includes the travelling salesperson, which asks, "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?"

# "HELLO, WORLD!"

Everything you need to know about our computing and digital making magazine for educators

## Q WHAT IS HELLO WORLD?

**A** Hello World is a magazine for computing and digital making educators. Written by educators, for educators, the magazine is designed as a platform to help you find inspiration, share experiences, and learn from each other.




## Q WHO MAKES HELLO WORLD?

**A** The magazine is a joint collaboration between its publisher, Raspberry Pi, and Computing at School (part of BCS, the Chartered Institute for IT).

## Q WHY DID WE MAKE IT?

**A** There's growing momentum behind the idea of putting computing and digital making at the heart of modern education, and we feel there's a need to do more to connect with and support educators, both inside and outside the classroom.

## Q WHEN IS IT AVAILABLE?

**A** Your 84-page magazine is available three times per year.

## IT'S FREE!

Hello World is free now and forever as a Creative Commons PDF download. You can download every issue from **helloworld.cc**. Visit the site to see if you're entitled to a free print edition, too.

# WANT TO GET INVOLVED?

There are numerous ways for you to get involved with the magazine. Here are just a handful of ideas to get you started

- **Give us feedback**
  Help us make your magazine better – your feedback is greatly appreciated.
- **Ask us a question**
  Do you have a question or a bugbear you'd like to share? We'll feature your thoughts and ideas.
- **Tell us your story**
  Have you had a recent success (or failure) you think the wider community would benefit from hearing about? We'd like to share it.
- **Write for the magazine**
  Do you have an interesting article idea or lesson plan? We'd love to hear from you.

## GET IN TOUCH

Want to talk? You can reach us at:
**contact@helloworld.cc**

## FIND US ONLINE

**www.helloworld.cc**

@HelloWorld_Edu

facebook.com/HelloWorldEduMag


helloworld.cc